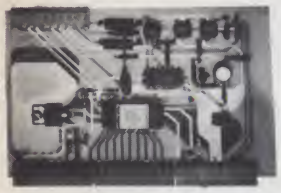
The Small Computer Magazine

kilobaud

Understandable for beginners . . . interesting for experts

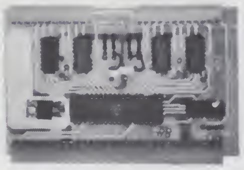
February 1978 / Issue #14 / $2.00 / DM7,50 / Sfr8,10 / Ffr16,0

PUTER SYSTEM

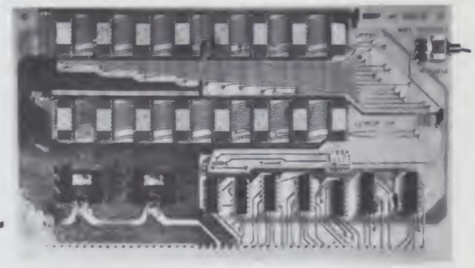


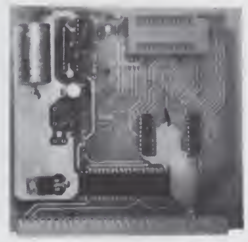
MP-S Serial Interface Kit
$35.00

MP-LA Parallel Interface Board Kit
$35.00

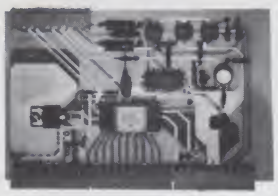


MP-8M Memory Board Kit. . . .$250.00
MP-16 Memory Board (assembled & tested). . . .$400.00





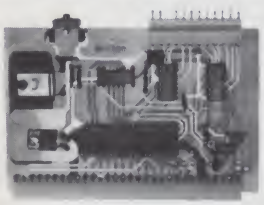
MP-R 2716 EPROM Programmer Kit
$44.95



MP-C Control Interface Kit
(Serial). . . .$40.00

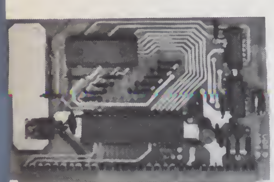
AC-30 Audio Cassette
Interface Kit. . . .$79.50

MP-T Interrupt Timer Interface Kit
$39.95



MF-68 Floppy
Disk Kit. . . .$995.00

6800 SOFTWARE
$4.95 & 9.95



MP-N Calculator Interface Kit
46.50

Please send item(s) checked. (All prices ppd. Continental U.S.)

— MP-6800	— CT-VM	— MP-C
— MP-6800/2	— MP-S	— AC-30
— GT-61	— MP-LA	— MF-68
— PPG-J	— MP-8M	— MP-T
— PR-40	— MP-16	— MP-N
— CT-64	— MP-R	

☐ Enclosed is $__________ or BAC #__________________________
or Master Charge #________ Bank #______ Expire Date________
Name____
Address____
City____________________ State ____________ Zip__________
☐ Please send catalogue of complete 6800 Computer System products.

SwTPc SOUTHWEST TECHNICAL PRODUCTS CORPORATION, Box 32040, San Antonio, Texas 78284

PUBLISHER'S REMARKS

Wayne Green

Saving Those Bits

The big computer systems have uninterruptible power supplies. Why haven't I seen a comparable supply for microcomputers? The closest thing so far is the memory-saver system pioneered by Bruce Seals with his memory boards . . . which is first rate.

Sure, hobbyists can live with an occasional power failure . . . but not comfortably, It's discouraging to see the lights start to flicker when you have a program running that took several hours to enter . . . and no copy on disk or tape . . . or perhaps the wall plug is accidentally pulled. These things happen.

Every now and then, one of the parts suppliers who advertise in the back of *Kilobaud* has some good rechargeable batteries available. I'd like to see a microcomputer power supply that includes batteries.

Over a year ago, I was promised an article on how to stop a computer . . . a discussion of hardware and software systems to shut a computer down with no loss of programs when the power goes off. The article never materialized, so readers still need information on this aspect of computing. Would any writers like to tackle that?

Backgammon Anyone?

One of the expected offshoots of microcomputers has been a sophisticated group of microcomputer games. With the appropriate program any microcomputer can play just about any game, as we know. With the cost of hardware going down, the next logical step was to build a complete computer into a popular game . . . and now we have one.

Texas Micro Games of Houston has a very nice little backgammon game with the computer built into the board. Backgammon aficionados may be interested to know that the computer plays a good, if not brilliant, game. But then, the prospect of being beaten most of the time by a little black chip could be quite a put-down.

TMG solved the problem of generating truly random dice throws (oh, yes, it throws the dice for you) by using the length of time you hold your finger on the button as the timing control. When you've just bounced a man off the board and you have one very vulnerable spot on the entry board, anything other than truly random dice throws would quickly bring on paranoia. However, the computer does play an honest game.

If you know how to play backgammon you'll be swinging along with this computerized version in about two minutes. The computer does not take foolish chances . . . and it does not miss any opportunities to raise hell with you when you open yourself up for mayhem. You'll enjoy it and you'll learn to play a good game . . . or lose a lot.

Simply push the spots on the board to roll the dice . . . the numbered spots move the men. If you make any mistakes in moving the board will light up, tilt and tell you what error you've made. Nothing can go wrong . . . go wrong . . . go wrong. . . .

For about $150, you have a backgammon partner that will beat you if you play anything but a tight game. You can't fool it by counting wrong . . . or by moving a marker when it isn't looking. Now, I still need an electronic cribbage partner.

The KB News

One of the ways we push for advertising in *Kilobaud* is by way of a newsletter sent out to manufacturers, computer stores and computer clubs. This letter is free and contains news of what is going on in the business, some advice on how to run businesses more profitably . . . and a lot of opinions by me.

If you are thinking of getting into the manufacturing or selling of microcomputers, you should make sure that you get on the newsletter mailing list. Computer-club secretaries should also sign up . . . there are some ideas every now and then on how to make clubs more effective . . . and you can get some interesting info on the industry.

All new stores should be sure to get monthly bundles of *Kilobaud* and keep stocked with our computer books. Magazines and books represent a substantial percentage of your profits, so make sure you have a good supply of them on hand.

Minimum Wage and Microcomputers

With the minumum wage soon going up to $2.65 per hour . . . and then to $3.35 per hour . . . the pressure will be on to put computers to work. Put yourself in the position of an employer and you can see the handwriting on the wall. Computer-store owners and salesmen please take note.

At $2.65 per hour, a weekly (40 hours) salary amounts to $106. So what? How much is that a month? At 4.33 weeks per month, that comes to $458.98 per month. Sure, we have inflation, and wages are going up. But let's see what we can buy in a computer system that will cost about the same as an employee making the minimum wage. It comes to $18,360 worth of computer on a five-year-lease basis.

The handwriting tells me that businesses are going to use computers and automation if they are going to keep costs down and be competitive—and this means a big need for microcomputers.

At the coming $3.35 minimum wage, we are into $134 per week, or $580.22 per month—the equivalent of a five-year computer-system lease for $23,210. That is a *lot* of computer. Considering the price reductions we see ahead on equipment, that is a fantastic amount of computer.

OK . . . you're smart. You see what's coming—you are going to plan ahead and be ready for it, right? There are some areas where little has been done to integrate microcomputer systems with business . . . such as speeding up the input of data. I haven't seen much being done yet with optical character recognition (OCR), product-code reading or other types of simplified input.

For example, what type of equipment would it take to automate mail orders for books, tools and small parts? Perhaps an order blank with spots to be marked that could be read optically . . . holes to be punched out with a ball-point pen . . . pencil or pen marks across a magnetic strip? Think about this and see what you can come up with. Whatever it is, if it works well, and is simple, you should be able to make a bundle. Reading orders for individual items isn't that difficult, but what about the customer's name and address? If you have to have someone key in that information, you have an expensive bottleneck. Have fun.

The more ways you can think of for businesses to save on labor, the more equipment you will sell.

Actually, the above calculations are understatements because employers have to match part of the taxes withheld, and provide insurance and other benefits. Also, you must consider the time spent for coffee breaks, sick time, vacations, etc. Your new employees may equal the cost of a $50,000 computer system.

Why the Narrow Columns

Perhaps you've wondered why *Kilobaud* has four columns of type per page while other magazines have three . . . or even two.

The human brain is a lot like a computer . . . or vice versa. You have to refresh the dynamic memory in your computer every now and then if you want to keep the memory perfect. The same thing goes for the human brain. If we want to keep memories accurate, we have to refresh them periodically. The four-column format in *Kilobaud* is designed so you can speed-read for memory refreshing. Wider columns tend to be more difficult to speed-read; the eye must jump back and forth. Narrower columns let your eyes run down the center of a column and take it all in easily. I'll explain how the speed-reading/memory-refresh technique works.

The first step is a careful reading of the material to be memorized. Make sure you understand it and follow the logical development of the article. Then, when you are done, go through it again, making sure you have the details you want in mind. Next, put down the book or magazine and sit back . . . relax . . . and go through the material right from the beginning in as detailed a fashion as

(continued on page 21)

EDITOR'S REMARKS

John Craig

Business Systems Security

You can pick up just about any of the major commercial computer publications (or your local newspaper) these days and find stories about computer crime. Various forms of embezzlement and fraud using computers have become very profitable—and difficult to detect. We usually hear about such cases in connection with multimillion-dollar systems in financial institutions and large corporations, but we need to start thinking of system security for small micro-based business systems. Since we're in the infancy stages of small-business-systems software development, *now* is definitely the time.

Computer crimes have been committed by system operators, programmers, managers, system analysts and just about anyone (even long distance via telephone) who can gain access to a system. People are not only stealing . . . I recently heard a couple of stories about disgruntled employees destroying company mailing lists and records upon being discharged. The companies went down the tube because of those vindictive actions.

It's doubtful such actions could ever destroy large corporations and institutions that are usually victims of computer crime . . . and they can usually absorb the losses. However, smaller businesses around the country that will soon have computer systems are going to be faced with the same kind of problems in the years to come. There are two important reasons why this point should be addressed now, before things get out of hand.

First, I'm sure that software system designers want to give their customers something that, quite frankly, will do the job . . . and something they can both be proud of. On the other side of the coin, if a lot of horror stories start cropping up about how entire small businesses were wiped out as a result of computerization . . . we're going to have a problem!

There is a *potential* problem here . . . it should be addressed (as a matter of fact, I'd like to see some articles on the subject) . . . and now is the time.

"Local" Happenings

I wonder if anyone knows where Lompoc, California, is? Probably not; but you'd be amazed at some of the personal computer activity that takes place in this little out-of-the-way town. I walked into the November 16th meeting of the Micro-8 Computer Club and was absolutely amazed by the hardware and software on display. Let me share some of it with you.

I walked in, and my arm was grabbed by Pete Bickerdike. "Come here," he exclaimed, "you gotta see this!" Pete is the western representative for Vector Graphic, and the display in Photo 1 is what he hauled me over to see. It took me a few moments to realize that I was looking at a digitized photograph of the house in the photo. Incredible! It turns out that Pete had a prototype of a new graphics board from Vector Graphic running in his machine. The board will be available sometime toward the beginning of the year and will sell for around $200.

The digitized photograph was obtained from a large time-share system. Vector Graphic is currently experimenting with slow-scan TV techniques for putting digitized images into memory. Such a system opens up all kinds of areas for system development. One of the first that comes to mind (because of the house) is a possible real-estate system that would have digitized photos of all of a realtor's offerings on a disk . . . and could be called up for display in a matter of one or two seconds.

There are countless other systems in which this kind of information retrieval could be put to use—don't forget *computer portraits.* For those of you interested in the "input" hardware and software for such a system, let me direct you to recent issues of *73 Magazine* and the series of articles by Clay Abrams that describe how he generates slow-scan TV pictures with his 6800 system.

Speaking of *73:* Back in September 1976, they ran an article describing the construction of and modifications to a surplus graphics terminal. The article was written by Steve Ciarcia (who is turning out a monthly column for *Byte* these days) and C. L. Robertson. I finally saw one of those terminals in operation at that meeting. The terminal had been interfaced to Jon Wiggins's Digital Group system, with some of Sublogic's 3-D graphics software patched in by David Bryant (see *Kilobaud* No. 10, page 50). They had some beautiful graphics up and running.

Also doing things at that meeting in Lompoc were a PET and a TRS-80 system (Photos 2 and 3). To say they attracted a lot of attention would be the understatement of the day. The

(continued on page 21)

Photo 1. Finally . . . some graphics from Vector Graphic!

Kilobaud is published monthly by 1001001, Inc., Peterborough NH 03458. Subscription rates in the U.S. and Canada are $15 for one year and $36 for three years. In Europe: Kilobaud erscheint montatlich bei Fachzeitschriftenvertrieb Monika Nedela, 7778 Markdorf, Markstr. 3. Abonnement DM 70.- plus Porto 7.20 + gesetzl. MWST; Sfr. 81.- plus Porto 7.20. Please write for other foreign rates. Second-class postage paid at Peterborough NH 03458 and at additional mailing offices. Publication No. 346690. Phone: 603-924-3873.

INCLUDE OLD ADDRESS AND ZIP CODE WITH ADDRESS CHANGE NOTIFICATION.

AROUND THE INDUSTRY

John Craig

Shugart Associates

Double-sided, double-density minifloppies have arrived on the scene! One of the biggest (perhaps the *only*) objection to minifloppies has been their "limited" capacity of 110 kilobytes (unformatted). Without digging out your pocket calculator you will be able to quickly deduce (as I did) that doubling the density and recording on both sides will quadruple the capacity of these little beasts (to 440 kilobytes).

That should take care of most objections, right? Only thing is . . . the double-density, double-sided techniques are also being applied to the standard-size floppies. They normally have a 400-kilobyte capacity, which will, strangely enough, be increased to 1600. The minis look better, but the standard-size looks *even* better! Without a doubt, one of the most desirable items for any personal system today is floppies.

I recently took a trip to San Francisco and dropped in on Shugart Associates for a visit. I got the impression that they would like very much for you to buy one of *their* floppy systems when you go out shopping. From the looks of their rather incredible manufacturing facility, I think they'll be able to deliver them, too.

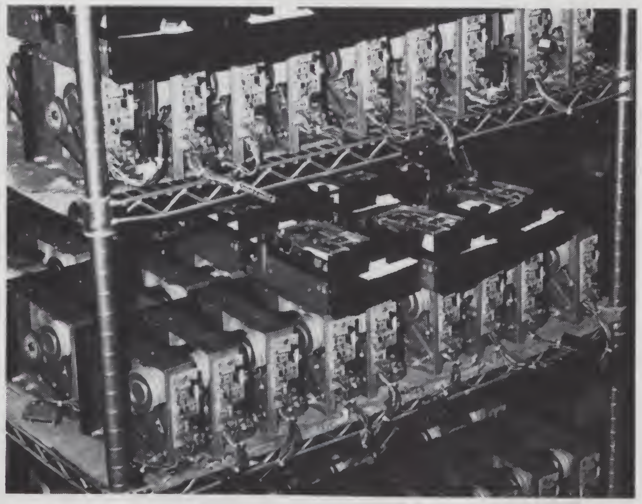


Photo 1.

The first thing to greet my eyes at Shugart was racks, and racks, and more racks filled with floppy drives. Even though I have a floppy system, I find that I drool uncontrollably when confronted with them. (Sometimes I think I'm more of a hobbyist/enthusiast than an editor.) Photo 1 is an example of the torment I was subjected to. (The security in the plant is fantastic . . . every time I tried to sneak a drive out, they caught me.)

The Shugart plant is an overwhelming 60,000 square feet in size. They're currently constructing another facility, next door; the new plant will add an additional *90,000* square feet! There aren't too many companies in this field experiencing those kinds of growth pains!

The "assembly lines" at the plant aren't really set in a conventional manner. Each assembler receives the parts necessary to build a complete unit via the conveyer belt shown in Photo 2 (a box of parts is shown being automatically diverted to a station needing parts). Completed units are placed on the bottom of the belt (underneath the one shown) and sent to quality control. From QC they are sent to final test and burn-in (shown in Photo 3). Here they get a 24-hour checkout and exercise, courtesy of a DEC PDP-11 . . . then back to QC for a final check before being shipped.

Shugart Associates was founded in 1973 by ten former employees of Memorex and IBM. Their objective was to become the largest independent (non-IBM) manufacturer of floppy-disk drives. I think they've done it. In their first year of operation they made about one million dollars . . . and went into the red. Growth has been steadily upwards ever since then (along with profit) and sales for fiscal 1978 are projected to be in excess of $48 million.

My thanks to George Sollman, director of product management and Gary Yost, product specialist, for the hospitality they extended during my visit (especially since I arrived on a rather frantic Friday afternoon). Needless to say, they're interested in hearing from stores and OEM customers interested in carrying their drives.

Shugart Associates, 415 Oakmead Parkway, Sunnyvale CA 94086.

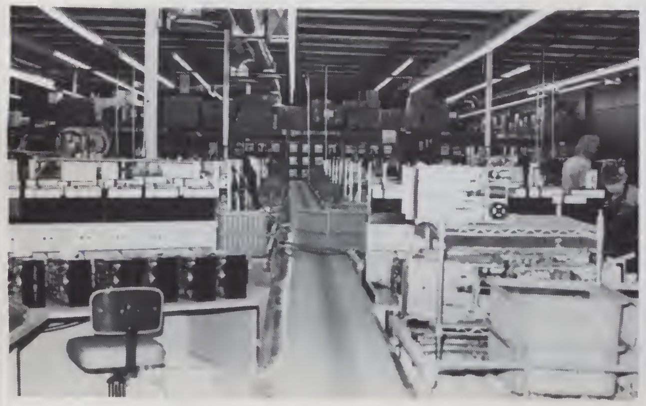


Photo 2.

Photo 3.

LEGAL/BUSINESS FORUM

Kenneth S. Widelitz
Attorney-at-Law

Warranty and Service Survey Results

The response to the *Kilobaud* Legal/Business Forum warranty-and-service survey was not exactly overwhelming. I suppose that the poor response is indicative of the industry's concern with warranty and service. That is to say, such concern is virtually nonexistent.

Although few surveys were returned, some interesting observations can be made based upon them.

First, not one of the warranties (I asked for copies from manufacturers who provide written warranties with their products) was prepared in consultation with an attorney. In fact, a few of the responses indicated that less than an hour was spent in developing the warranty policy and written warranty. As a result, you won't be surprised to hear that not one of the warranties conforms to requirements of the MMWA or SBCWA (see the two previous Legal/Business Forums).

Retailers

About 20 percent of responses were from retailers, who indicated that they performed an average of 20 hours per week of in-warranty service. Hourly charges for in-warranty service ranged from $5 to $15. The hourly charge for out-of-warranty repairs was reported to range from $10 to $20. With the exception of remibursement for defective parts, responses indicated retailers were not reimbursed by manufacturers for performing in-warranty repairs.

The retailers reported that their average turnaround time for making in-warranty repairs was one to two weeks. They reported that turnaround time was three to four times longer when the product was returned to the manufacturer for repairs. As a final observation on retailer response: Written warranties were not made available prior to sale. That means noncompliance with the Presale Availability Rule.

Manufacturers

About half of the responding manufacturers reported that they provided "full" warranties. However, those manufacturers also indicated that they do not pay round-trip transportation for goods returned to them for in-warranty repairs. As you will recall, in order for a warranty to be full, the warrantor must remedy the defect(s) *without charge*. Thus, some manufacturers who believe they provide a "full" warranty actually do not do so. One manufacturer who stated that a "full" warranty was provided with his products indicated that the warranty extended only to the original purchaser. The MMWA requires that a warranty, to meet full-warranty qualifications, must extend to any person who is a "consumer" of the product. Therefore, such a limitation is not compatible with a full warranty. I found one response most amazing: A manufacturer who claimed he was giving a "full" warranty stated, in response to a question on the survey, that his warranty did not limit incidental or consequential damages. Yet, such a limitation appeared on the face of the warranty. From that you might conclude that some manufacturers don't know what is contained in their warranties.

Regarding the duration of the warranties: they ranged from 30 days to 180 days—on kits and

(continued on page 22)

THE BASIC FORUM

John Arnold/Dick Whipple

BASIC Forum begins this month with fewer correspondents than last time, but the subjects remain as interesting as ever.

Data Storage

Several columns ago, we considered the storage of data via tape and disk. Two letters received recently refer specifically to tape storage.

The first comes from Gene Embry, 1325 Helmsdale, Cary NC 27511. Gene has devised a technique for storing data using his SWTP 6800 system. We regret that his program is too long to publish, but we will attempt to describe it well enough so that you can try your own. (Or write Gene directly—that is another communication channel we want to open!)

Initially, a portion of memory is reserved for data storage. As data is entered via the keyboard, it is converted to a string, then stored character by character in the previously reserved memory using the POKE command. When all data is stored, the reserved area of memory is written on the cassette tape using the MIKBUG tape utility routines. Presumably, the reverse of the process is used to reinstall the saved values into the BASIC program.

In an earlier Forum we described a tape storage technique wherein BASIC was used to write directly to tape. We mentioned then that timing problems could arise due to the processing delays inherent in BASIC. Gene's method avoids this problem by using memory as an intermediate storage medium between BASIC and the tape. Thus, the tape dump proceeds at its normal rate and maximum data density, for the given baud rate is achieved.

A second letter on the subject of tape data storage comes from Bill Roch, 24000 Bessemer Street, Woodland Hills CA 91367. Bill's company manufactures a tape system that can be adapted to different BASIC interpreters.

"In the August Forum (p. 15), comments were made about lack of BASIC I/O for GETting and PUTting records from and to cassette tape. Regardless of the statements used, it still gets down to reading and writing records.

"RO-CHE Systems supplies a BASIC I/O driver with their Multi-Cassette Controller that does just that. BASIC is patched to the I/O drive, which then handles the I/O, either to the console or the Cassette Operating System, which can handle up to four cassette recorders.

"The Cassette Operating System opens and closes files; even named files can be used. File marks are written and read for start of file and end of file. The operating system also takes care of turning the cassette recorder off and on.

"When a logical record is written, the blanks are compressed out and it is stored in the output buffer. As records are written, the output buffer fills up. When full, it is written to the cassette tape. To read a record, the 256-byte physical record is read from cassette and placed in the input buffer. With the blanks put back in, each read brings a logical record up for processing until the buffer is empty, when another read from the cassette takes place.

"The kit includes software to handle these operations, and an assembler and patches for assembling large programs from tape. Also, a *BASIC* demo program and a data record file are included.

"The RO-CHE Multi-Cassette Controller plugs directly into a Tarbell cassette interface board, so the reading and writing is fast. Application software for sorting, merging, copying, reformating, text editing, etc., is optionally available."

Numerical Data

In the September Forum, we

published a beginner's hint concerning ways to columnize numerical data. Two letters concern this topic. A frequent contributor, James Caldwell, 3120 NW 61st Terrace, Oklahoma City OK 73112, writes:

"Reference George Haller's use of LOG base 10 to columnize decimal figures. Some BASICs do not allow the use of LOG base 10, and the same results can be obtained with only seven lines versus 11 lines for use with money values by right-hand justifying with the STR$(A) and LEN(A$) statements in SWTP 8K BASIC. LEN(A$) returns the number of characters contained in A$, and STR$(A) returns the string value of the numeric value A. By converting the numeric value to a string value with STR$(A), and then determining the length of A$, we can start printing that number of spaces before the desired decimal point (+2)." (See Example 1.)

It should be noted that Jim's program correctly handles the last two data values because of the DIGITS=2 statement contained at the beginning of the program. This SWTP BASIC statement causes two decimal places to be printed, regardless of the presence or absence of a decimal fraction. Without the DIGITS statement (which is generally the case), Jim's method would not be entirely equivalent to George Haller's. Using the LEN and STR$ functions establishes columns by left-justifying the numeric values. The LOG base 10 approach columnizes by alignment of the decimal. Example 2 gives side-by-side results to illustrate this point.

We also noted that George's method did not provide the trailing zeros necessary for printing dollars and cents. For example, in printing ten dollars we would get 10, not 10.00, as required. Our next letter appears to solve the problem—at least for owners of SWTP BASIC. Charles Knott, 3008 Congress Rd., Camden NJ 08104, writes:

"Using an MSI computer with either of SWTP's 8K BASICs (both by Bob Uiterwyk), here's a program to fill in the missing zeros." (See Example 3.)

Once again the DIGITS statement in SWTP BASIC comes to the rescue. Without it, the problem remains. You might take a few moments to tackle the problem; i.e., use STR$, LEN, LOG without DIGITS to columnize and print full dollars *and* cents. If you come up with something, drop us a line.

The prime-number problem in the September Forum continues to generate interest. In the last Forum, we summarized as many responses as we had received by that date. Additional letters have arrived, and we want to acknowledge them. Unfortunately, space requirements don't permit publishing complete details. Instead, we have compiled a name-and-address list of those submitting solutions along with the type of computer each used (see Table 1). Thanks to those who contributed. We hope you will do so again!

BASIC Test Programs

Our final letter introduces a topic that we hope will stimulate contributions to the Forum. It comes from Robert J. Lurie, 8 Tingley Rd., Morristown NJ 07960.

"I am a newcomer to computers, but I now own a SOL-20 and am in the process of rewriting, in SOL BASIC, many of the application programs I have developed over the years for use on a variety of programmable calculators. Many of these programs involve numerical integration and other operations for which the BASIC statement in Example 4 appears ideally suited. These programs usually bombed!

"You might be interested in the attached program called A BASIC TEST (Program A). The SOL BASIC 5 interpreter scores ~ (approximately) 25 percent (~50 percent if D=4). Of course, anything less than 100 percent must be considered a failing grade—at least for reasonably small values of D.

"Naturally, for any FOR-NEXT structure there exists an equivalent IF-THEN structure. However, having been stung bad-

```
 5 DIGITS=2:REM SET UP TWO DECIMAL PLACE PRINTING
10 READ A
20 IF A< 0 THEN STOP
30 PRINT TAB(8-LEN(STR$(A)));A
40 GOTO 10
50 DATA .01,.12,1.23,12.34,123.45, 1234.56,1.2,10
60 DATA -1
 RUN
     .01
     .12
    1.23
   12.34
  123.45
 1234.56
    1.20
   10.00
 STOP AT 20
```

Example 1.

LEN + STR$ (No DIGITS=2)	LOG base 10
.01	.01
.12	.12
1.23	1.23
12.34	12.34
123.45	123.45
1234.56	1234.56
1.2	1.2 Note
10	10 difference here.

Example 2.

```
0001 REM /// SUBSTITUTE FOR PRINT USING STATEMENT ///
0002 DIGITS=2
0003 REM /// M.S.I & SWTP 8K BASICS VER 1&2
0004 REM /// WHAT HAPPENED TO THE ZERO's—KB 9/77 C.KNOTT
0005 K1=LOG(10)
0010 READ A0
0020 IF A0< 0 THEN STOP
0025 REM /// LOG (x) BASE 10 = LOG (X) BASE E/LOG (10) BASE E ///
0030 IF LOG (A0)/K1< 0 THEN 60
0040 PRINT TAB (6-INT(LOG(A0)/K1));A0
0050 GOTO 10
0060 PRINT TAB (5);A0
0070 GOTO 10
0080 DATA .01,.05,.10,.25,.50,1.00,1,12.00,12,123.78,1234.,12345.00
0090 DATA .1,.5,.8
0100 DATA -1

RUN
     0.01
     0.05
     0.10
     0.25
     0.50
     1.00
     1.00
    12.00
    12.00
   123.78
  1234.00
 12345.00
     0.10
     0.50
```

Example 3.

```
FOR I=F to L STEP (L-F)/D   (D=integer)
. . . .
NEXT I
```

Example 4.

ly by BASIC 5's lousy FOR-NEXT-STEP performance, I decided I'd better test to see how well it does with IF-THEN statements (i.e., are they also demanding more precision than is available in the selected word length?). Hence, Program B (ANOTHER BASIC TEST). For random Ds between 2 and 9 (integer values only), BASIC 5 scores a rotten 60 percent. Another failure! (Only if D is a small power of 10 does it work.)

"One can write around these interpreter deficiencies, but only at the cost of increased program complexity and length."

Interpreter-deficient computers that can't add or accurately complete a simple FOR-NEXT loop! Surely Robert must be feeding his micro the wrond kind of electricity! Not so fast—it is possible that he has simply stumbled onto a weakness in the way BASIC handles certain arithmetic operations. The culprit here is well known to experienced programmers and is commonly referred to as round-off error. Let's delve into this problem.

Round-off Error

In many programming applications, we use small whole numbers, and we don't encounter any difficulty since whole numbers can be represented ex-

```
 10 REM (A BASIC TEST)
 20 RANDOMIZE: REM may not be needed with your interpreter
 30 C=0
 40 FOR J=1 to 100
 50   F=RND(1)
 60   L=RND(1)
 70   IF F=L THEN 70: REM a bit of nit-picking!
 80   D=INT(2+9*RND(1)): REM GENERATES INTEGER BETWEEN 2 and 10
 90     FOR I=F to L STEP (L-F)/D
100       IF I=L THEN C=C+1
110     NEXT I
120 NEXT J
130 PRINT "SCORE=";C;" percent"
999 END
```

(The heart of this program is in lines 100-120.) (The "J" loop works OK.)

Program A.

```
 10 REM (ANOTHER BASIC TEST) This program tests lines 90 and 100
 20 RANDOMIZE: REM not needed in "BASIC 5"
 30 C=0
 40 FOR K=1 to 100
 50   F=RND(1)
 60   L=RND(1)
 70   IF L=F THEN 60 REM one chance in 10^8!
 80   D=INT(2+8*RND(1)) REM: any integer between 2 and 9
 90   FOR J=0 to D
100     I=F+(L-F)*J/D
110     IF I=L THEN c=c+1
120   NEXT J
130 NEXT K
140 PRINT "SCORE=";c;" percent"
999 END
```

Here the loops are OK. It is interesting to watch what is happening; change:

```
110 IF I=L THEN c=c+1: GOTO 120
115 IF J=D THEN PRINT L,I
```

Program B.

Ed Juge 2000 Thousand Oaks Dr. Burleson TX 76028	Radio Shack TRS-80
Robert B. Pierce 2900 U.S. Steel Building Pittsburgh PA 15219	SBC CALL 370-Time Share
Ross Cooling 299 Forman Ave # 14 Toronto Ont, CAN M45 256	Imsai, TDL ZPU
D. Vitanov 404 Attica Dr. SE Calgary Al, CAN	SWTP 6800
D'Arcy Roberts/Gordon Thomas 660 Laurier Blvd. Brockville Ont, CAN K6V 5X8	PDP-8E
Greg Oliver 256 Stone Rd. Rochester NY 14616	H-P 2100
John K. Jordan 90 Outer Dr. Oak Ridge TN 37830	SWTP 6800
C. E. Eakin 6932 S Niagara Ct. Englewood CO 80110	IBM 360/30 (FORTRAN)
Anne Kohlstaedt 5947 E. 82nd St. Indianapolis IN 46250	PT SOL-20
William Duncan, Jr. Penthouse Four 30 Godstone Rd. Willowdale Ont, CAN M2J 3C6	TI-58 calculator
Stephen T. Klock Byte Shop of Arizona 813 N. Scottsdale Road Tempe AZ 85201	Imsai
Michael Powers 3135 NE Lancaster St. Corvallis OR 97330	CDC 3300
D. Kinoshita/B. Sinfield 8902 13th St. Edmonton Al, CAN T6B 2H2	Imsai
Joe Holliday Box One Luverne AL 36049	H-P 2000
Murray W. Diller P.O. Box 280 Longview WA 98632	Alpha Micro
Patrick Allocca 80-16 88th Rd. Woodhaven NY 11421	SWTP 6800

Table 1.

actly up to the precision limit of the BASIC software (usually 6 to 7 decimal digits). Thus, structures like Example 5 seem to work fine; when the loop is finished, I has the expected value 100. In Robert's first program, he deviates from this routine practice and sets up a FOR-NEXT loop with various fractional limits and STEPs. When a fraction is converted to binary, it may not have an exact representation within the precision limits available. Thus, the value stored in memory may fall short of the original decimal value.

Consider another, more familiar, conversion process—that of changing a common fraction to a decimal fraction. The common fraction 1/8 converts exactly to the decimal fraction 0.125. Adding the decimal value eight times gives the expected value, 1.

Now consider the fraction 1/3. It cannot be representated exactly, no matter how many digits we wish to use. If the precision limit is set to three, we would have 1/3 = 0.333. By ignoring the digits beyond the third place (rounding off), we necessarily introduce a small error that is difficult, if not impossible, to correct. For example, if we add 0.333 to itself three times we get 0.999, not 1. In fact, each time we add 0.333, the error grows larger and more unacceptable. One way to stave off the effect of the accumulating error is to use greater precision; i.e., represent 1/3 as 0.333333. This permits more addition operations before the error grows uncomfortably large.

The decimal-to-binary conversion process works much the same way. Decimal fractions may not convert exactly within the precision limits imposed. The binary digits beyond those allowed are merely ignored. Thus, the value stored may differ slightly from the original value. When the stored value is subjected to numerous operations like addition, the small error may accumulate to a noticeable, and perhaps objectionable, size. In Robert's first program, the round-off error in the STEP value may give a final index value different from that imposed in the original FOR limit. In this case—Glitch! Tilt!

To illustrate round-off error in number conversion, we wrote a simple program and executed it with two different BASICs: Mits Disk Ver. 4.1 and Disk BASIC ETC. The former uses binary arithmetic; the latter, Binary-Coded Decimal (BCD). BCD-based interpreters (like SWTP BASIC and BASIC ETC) use no binary conversion but store the arithmetic values in four-bit groups corresponding to the digits of the original number. Although used extensively in calculators, BCD is not generally found in high-level language processors since it requires more memory and has slower execution times. The listing and results appear in Example 6.

The binary conversion process in Mits BASIC produces the small round-off error shown. The BCD routines in BASIC ETC track the FOR-NEXT without error. Note that the BCD arithmetic was slower in execution.

```
FOR I = 1 to 100 STEP 2: NEXT I
```

Example 5.

```
 5 A = 0
10 FOR IO = 1 to 10 STEP .001
20 IF A = 1000 THEN PRINT IO:A = 0
30 A = A + 1
40 NEXT IO
50 PRINT IO
99 END
```

Mits	BASIC ETC
2.00005	2.
2.99997	3.
3.9999	4.
4.99983	5.
5.99976	6.
6.99968	7.
7.99961	8.
9.00002	9.
10.00004	10.
RUN TIME = 1:28 min	= 1:43 min.

Example 6.

```
10 FOR IO = 1 to 100
20 IF IO/10 = INT(IO/10) THEN PRINT IO;
30 NEXT IO
RUN
10 20 30 40 50 60 70 80 90 100
```

Example 7.

```
10 FOR IO = 1 to 10 STEP .001
20 IF IO = INT(IO) THEN PRINT IO;
30 NEXT IO
99 END
RUN
1
OK
```

Example 8.

```
110 IF ABS (I-L)< 1E-6 THEN c = c + 1.
```

Example 9.

When round-off errors are present, programming techniques have to be modified. For instance, printing every tenth value in a FOR-NEXT loop is often done with the trick shown in Example 7. Only when IO is 10 or a multiple of 10 will the equality be true and IO printed. Unfortunately, if we try to rewrite our test program using this technique with Mits BASIC, we run into trouble (see Example 8).

What happened to 2, 3, . . . etc.? Remember: With round-off error, the equality may never be true! You see now why we had to use the counting variable A in our test program. FORTRAN programmers are trained to avoid tests for equality unless working with whole numbers. BASIC programmers should also take heed.

It should also be noted that additional round-off errors can be generated by arithmetic operations like division. For instance, a non-even division will result in a quotient limited to the precision of the BASIC's arithmetic; any remainder is ignored.

Tossing away the remainder is a form of round-off error that introduces a small inaccuracy into the calculation. Errors of this type are not limited to binary-based interpreters, but exist whenever the division process exceeds the precision limit of the particular device being used (BCD-based interpreters and calculations included).

Referring again to Robert's letter, we need to clear up a point or two about his second program. Although he has identified a source of difficulty for numerical analysis with BASIC, we are not convinced that his program can correctly identify the magnitude of the problem for a particular interpreter. Our question arises in the structure of the program—in particular the use of the RND function to create the FOR-NEXT limits.

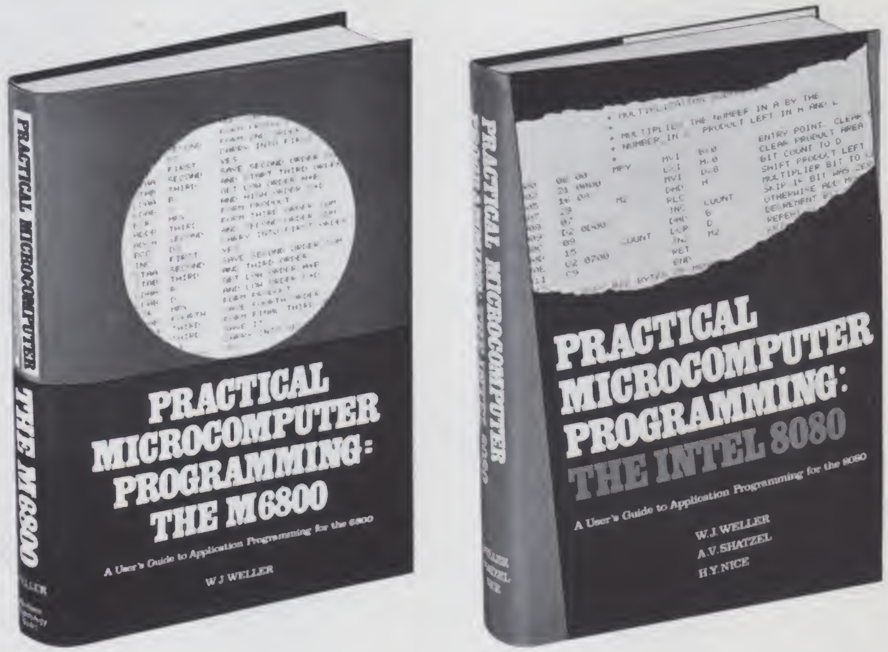
In a given iteration, RND generates a random number whose value extends to the limit of precision of the interpreter's arithmetic routines. When division is applied to these values (line 100 of his program), the quotient will not always be without some round-off error. The interpreter cannot be taken to task for errors of this type since they are fully expected. Instead, the programmer should anticipate such errors and use programming techniques that avoid undesirable results. For instance, it would have been more realistic in Robert's program to have demanded that the error remain less than some small value rather than tallying only exact equalities. Thus, we might change line 110 to read like Example 9.

The topic of error analysis deserves more consideration, and we are interested in your participation. Send along your comments and perhaps a program that you believe tests the number-handling capability of BASICs. We would like to follow Robert's suggestion and obtain execution data on his programs in other computers. Send your results with RUN time and output device to: The BASIC Forum, PO Box 7082, Tyler TX 75711.

If you need *real* results from your 8080 or 6800 based system

Then scan this list of topics . . .

- binary arithmetic
- logical operations
- organization of a computer
- *referencing* memory
- carry and overflow
- multiple precision arithmetic
- loops
- shifting
- software multiplication and division
- number scaling
- floating point arithmetic
- stack pointer usage
- subroutines
- table and array handling
- number base conversions
- BCD arithmetic
- trigonometry
- random number generation
- programming of the 6820 PIA
- programmed input/output
- control of complex peripherals
- programming with interrupts
- a software time of day clock
- multiple interval timers in software
- data transmission under interrupt control
- polling
- debugging techniques
- patching a binary program
- full source listing of a debug program . . .



Every one of these topics and many, many more are discussed in the **Practical Microcomputer Programming** books. In chapter after chapter and scores of formal program examples, the basic skills of assembly language programming are developed step by step. The examples are real and have been tested and proven. They run, and more important, they teach. If you're tired of generalities, reproductions of manufacturers data sheets and books with examples that don't run, then there is only *one* place to go, the **Practical Microcomputer Programming** series from Northern Technology Books. At $21.95 each they are the best bargain in programming information available anywhere.

*Order now . . . Start getting **real** results from your 8080 or 6800 based systems.*

K

Northern Technology Books Box 62, Evanston, IL 60204

- ☐ Practical Microcomputer Programming: The Intel 8080 $21.95
- ☐ Practical Microcomputer Programming: The 6800 $21.95

☐ check on US bank enclosed ☐ money order enclosed

Illinois residents add $1.10 state sales tax.
Foreign orders add air mail postage if desired (.8 kg).

Please type or print

Name____________________
Company____________________
Address____________________
City__________ **State**__________ **Zip**__________

Prepaid orders only

N8

NEW PRODUCTS

INFO 2000 Business System.

New Shugart Minifloppy Drive

Shugart Associates recently announced their SA450 double-sided/double-density/double-headed minifloppy drive. Related to the SA400 minifloppy, the SA450 reads and writes data on both sides of a minidiskette without removing it from the drive. This design uses two proprietary glass-bonded ferrite-ceramic read/write heads based on proven SA850 double-sided drive technology. The new SA450 drive will store up to four times the on-line data of the SA400, or 440K bytes unformatted.

The SA450 costs $450, about 25 percent more than the SA400. This latest double-sided floppy drive follows the introduction of Shugart's SA850 standard size double-sided drive, which was introduced in April 1977.

New double-sided 5.25-inch minidiskette media for the SA400 drive are available in the SA154 (soft sectored), SA155 (hard sectored, 16 hole) or SA157 (hard sectored, 10 hole). The double-sided minidiskettes are available in a box of ten from several diskette manufacturers for approximately $65 per box.

As with the standard minifloppy, an activity light, write-protect circuitry, and 35-track recording format are standard features. Data transfer rate is 125 or 250K bits/second. Other features include low heat dissipation (18.7 Watts or less), drive motor and 25 ms track-to-track access time.

Shugart Associates, 415 Oakmead Parkway, Sunnyvale CA 94086.

24 Channel Logic Analyzer Plugs into S-100 Bus

Utilizing Bipolar memory circuits, Databyte Inc., has developed a low-cost 24-channel logic analyzer capable of acquiring up to 256 × 24 data sets. The unit, called the Datalyzer, is designed for use in an S-100-type computer system and has the additional capability of self-monitoring the S-100 address and data lines by simply changing operating modes.

In the external mode, the Datalyzer can acquire data at a 100 MHz clock rate, has low-capacitance input probes and is designed for TTL logic levels. Probes are arranged in three groups of ten, each set containing a ground lead, eight data channels plus a clock or qualifier input.

In the internal (S-100) mode, the Datalyzer can perform the additional function of disassembling a running program exactly as the program was executed.

Triggering, display formatting, and operational modes of the Datalyzer are controlled from the user's system-input device. The trigger word can be up to 16 bits wide, with any combination of 1, 0 or X (don't care) states possible.

The Datalyzer uses equipment that the average S-100 system already has and does not require the use of an oscilloscope. The price in kit form is $495; the assembled unit is $595. Included are three probe assemblies, a system monitor on paper tape and an applications manual that can be purchased separately for $7.50.

Databyte, Inc., PO Box 14, 7433 Hubbard Ave., Middleton WI 53562.

INFO 2000 Business System

INFO 2000 Corporation of Carson CA has announced their new computer system for small-business data processing. The INFO 2000 Business System competes in performance and functional capability with minicomputer systems now selling for over $30,000. The complete system consists of a Z-80-based computer, dual flexible-disk drives, high-speed printer, video terminal and extensive business-applications software.

The mainframe employs the S-100 bus architecture and contains up to 56K of RAM memory, 8K of EPROM, a filtered forced-air cooling system and heavy-duty power supply.

Mass storage is provided with PerSci dual flexible-disk drives. The printer is a 160 cps, 132-column line device with all 95 ASCII alphanumeric and graphic characters including true lowercase letters with descenders. Printer capabilities include graphing and charting. The video console uses a commercial-quality keyboard with numeric keypad and displays all ASCII characters with easy-to-read lowercase letters. Other features include dual display intensity, protected fields and high speed at 19,200 bits per second.

All operating software is included. Business applications include CPA 2000 (accounting package), TEXT 2000 (a word-processing package) and full disk-operating system for software development. Optional software is available. The INFO 2000 Business System is priced under $10,000.

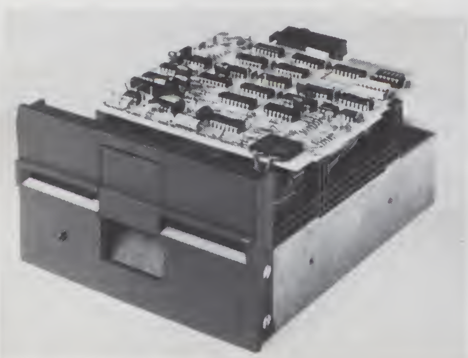
INFO 2000 Corporation 20630 S. Leapwood Ave., Carson CA 90746.

New S-100 Bus "Virtually Noise Free"

Peace and quiet has finally come to the S-100 bus after more than two years of static. A recently announced S-100 bus board in kit form produces "textbook-clean" signals. According to the manufacturer, Thinker Toys, the high signal quality of the new board is produced by a complex noise-control system called Noiseguard.

Noiseguard is the only bus board noise-squelching system currently available with both full shielding of signal paths and active termination of all data lines.

Signal isolation is achieved by a cross-coupled system of ground lines that are interlaced between



Shugart SA450 floppy disk drive.

Datalyzer logic analyzer.

signal lines and cross-coupled ground planes. This surrounds each signal in a cocoon of extremely quiet space to eliminate noise and cross talk between signal lines.

Furthermore, each data line is actively terminated by a circuit that absorbs signal reflections and noise. This low-power noise-absorption system is much simpler than other termination systems, utilizing a high-speed operational amplifier and amplifying transistors.

The Noiseguard two-way noise-squelching system is an exclusive feature of the Wunderbuss S-100 bus board from Thinker Toys, 1201-10th St., Berkeley CA 94710.

TSC 8080 Text Editor

The TSC Text Processing System allows the use of over 50 special text-formatting commands. Also included are capabilities for macro definition to define and build special formatting commands, number registers that can be used like variables in a program, conditional command execution, text diversion for later use (such as footnote processing), the ability to prompt a terminal for text during the formatting process, and a feature that allows sending informative strings to the terminal.

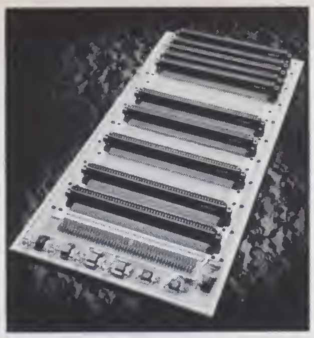
The program will also output numbers in Arabic, and capital or small Roman numerals. Tab columns may be defined as well as the tab character and the tab fill character. Environment switching is permitted for easy parameter changing, and a loop command is available for repeated formatting jobs such as form letters.

TSC recommends their Text Editing System be used in conjunction with the processor. These two packages will give the user all the powers of the most complete word-processing system. SL68-29 Manual and Source Listing, $32; with cassette, $38.95; with paper tape, $40.

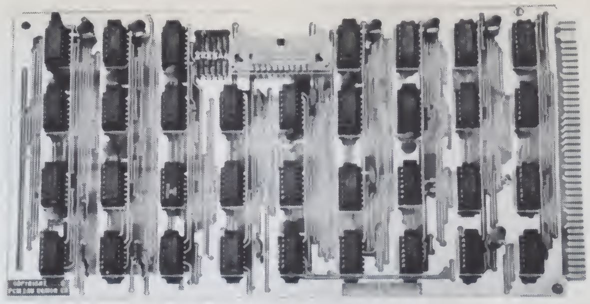
Technical Systems Consultants, Box 2574, W. Lafayette IN 47906.

Optical Reader for Paper Tape

The Byte Reader is an inexpensive yet versatile optical paper-tape reader designed to fill the needs of the computer hobbyist. It features an exclusive Lite Optimizer circuit, which senses the



Thinker Toys S-100 bus board.



PCM floppy-disk interface.

intensity of the external light source and automatically adjusts the sensitivity level of the photo transistors for proper operation.

The device features LED data bit indicators that enable you to visually verify data being sent to the computer. The kit sells for $69.95; fully assembled, $84.50. Include $3 for shipping and handling.

Microtronics, PO Box 7454N, Menlo Park CA 94025.

PC/M Dual Floppy-Disk Interface

Pacific Cyber/Metrix, Inc., has a dual floppy-disk interface module for its 12-bit microcomputer system, the PCM-12, which is software-compatible with Digital Equipment Corp.'s PDP-8 family of minicomputers.

PCM's floppy-disk module interfaces the PCM-12 to Data Systems Design's Model 210 floppy-disk memory system and sells for $295 assembled; $169 in kit form.

Plug-compatible with DSD's Model 210, PCM's 12440 floppy-disk module allows PCM-12 users to execute all PDP-8 floppy-disk diagnostics and makes the PCM-12 system fully compatible with all mass storage operating systems already developed for the PDP-8 minicomputers.

PCM, 3120 Crow Canyon Road, San Ramon CA 95483.

Digital Pulser Probe from CSC

The new DP-1 Digital Pulser from Continental Specialties Corporation features a bit of automation. Internal circuitry monitors the node being probed, then presets the dual-mirror output circuitry to pulse the node the other way. It delivers a strong enough pulse to kick most lines with no need to desolder, unplug or isolate, and the output is short-circuit protected. It can drive a dead short indefinitely with no danger of damage to the Pulser.

The Digital Pulser derives its power from the circuit being investigated to help assure logic-level compatibility, and a switch selects appropriate threshold levels to trigger either TTL or CMOS circuits.

For cleaner high-current pulses, a short ground jumper connects near the Pulser's probe tip. Probe tips are interchangeable with optional tips and accessories available from CSC. The power connector is a standard phono plug. The DP-1 Digital Pulser is priced at $74.95.

Continental Specialties Corporation, 44 Kendall Street, New Haven CT 06509.

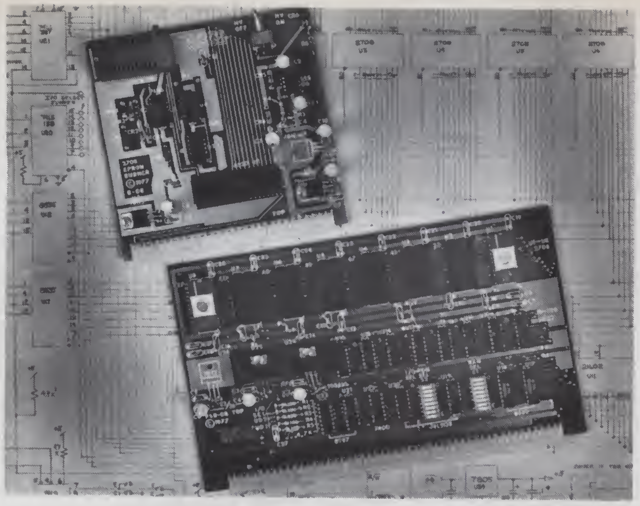
TSC 6800 Text Processor

The 8080 Text Editing System is both line oriented and content oriented in that specific lines can be referenced by a particular line number, an offset amount or by a string of characters contained within the line. The program resides in 5K of memory.

Such commands as PRINT, INSERT, DELETE, FIND, REPLACE and VERIFY are included. The current line pointer always points to the beginning of a line. There is auto line numbering, and the line numbers may be turned off. Pointer movers include TOP, BOTTOM and NEXT. Other features are TAB column set and character definition, OVERLAY, APPEND, HEADER and block MOVE or COPY. READ, WRITE and SAVE commands are available for use with tape systems, and an extensive CHANGE command allows one to change any or all specified occurrences of one string into another. ZONES may be set to allow column restriction of all string searches and replacements. Multiple com-

Microtronics' tape reader.

DP-1 Digital Pulser.



Micro Works B-08 (top) and PSB-08.

mands per line are permitted, and most commands are global in that they can operate over the entire file.

The 8080 Text Editing System sells for $28.50 and includes user's manual, source listing and hex dump listing. Paper tape is available for $9.

Technical Systems Consultants, Box 2574, W. Lafayette IN 47906.

The Micro Works 6800 Computer Accessories

PSB-08 PROM System Board. This board features 1K of high-speed (350 ns), low-power RAM and space for up to eight 2708 EPROMs. The exclusive I/O select feature allows the user to move the I/O locations up to any unused 1K block in the EPROM memory space. An optional +12 V regulator can be installed for systems incorporating the Smoke Signal Broadcasting PS-1 power supply or its equivalent. Prices: PSB-08 (EPROMs not included) $119.95; PSB-08R (regulated +12) $124.95.

B-08 2708 EPROM Programmer. The B-08 is a compact 2708 EPROM programmer that fits in a standard SWTP 6800 I/O slot. A safety switch and LED indicator provide control over the high programming voltage generated on the board. An industrial quality Textool zero-insertion-force socket and extended board height allow effortless PROM insertion and retrieval. Fully commented source listings of the Micro Works U2708 PROM Utility software (U2708/C000 or U2708/FC000—$29.95; U2708/1000—KC Standard 300 baud cassette tape—$9.95) are included, allowing quick and reliable programming and copying of 2708s. An optional +12 V regulator is available. Prices: B-08 $99.95; B-08R (regulated +12) $104.95.

UIO Universal I/O Board. This I/O board has space for a 40-pin wire-wrap socket into which you may plug either a 6820/21 PIA or 6850 ACIA; the data and control lines are connected to the appropriate edge connector pins. All other bus connections are brought out to a 16-pin socket pad. A +5 V regulator and all Molex connectors are provided; regulated +5 and ground are bused among the locations for up to 35 14-pin ICs. Price: $24.95.

X-50/X-30. These double-sided extender boards have bus extensions on the bottom and a ground plane on top. Both sides are soldermasked. Silk-screened bus pin designations and ground clip attachment points make debugging easy. Prices: X-50 (S-50 bus) $29.95; X-30 (S-30 I/O bus) $22.95.

All Micro Works 6800 computer accessories come fully assembled, tested and burned in as necessary. All software is fully source listed and commented; complete schematic diagrams are included.

The Micro Works, PO Box 1110, Del Mar CA 92014.

Three-Dimensional Graphics

The Sublogic 3-D micrographics package for microcomputers will allow a user to view two-dimensional perspective projections of three-dimensional scenes from any location in space. Driving and flying simulations, artistic projections, design projections, engineering analysis and advanced games are now simple and economical.

Two versions of the graphics package will be offered: a minimal subset BASIC version for general purpose, slow-speed graphics on any microcomputer system; the 6800 optimized assembly-language version with dynamic graphic capabilities for advanced simulation and complex graphics.

Adaptation instructions, program listings, applications, interface and testing information will be supplied with each package. The BASIC version will retail for $22. The 6800 package will be priced slightly higher.

Sublogic, PO Box 3442, Culver City CA 90230.

PROM Programmer

Oliver Audio Engineering has a new low-cost series of piggyback PROM programmers. The PP-2708/16 PROM programmer plugs directly into any 2708 or TMS-2716 memory socket.

The PROM to be programmed is placed in the zero-insertion-force socket and the data is dumped over the eight lower address lines using OAE's proprietary interface technique. No additional power supplies are required, and all timing and control sequences are handled by the programmer. Because of this interfacing technique, only a short software routine is required to give you the power of the most expensive programmers. In addition, multiple programmers may be connected in parallel for gang programming.

Each unit comes complete with a dc-to-dc switching regulator, ten turn cermet trimmers for precise voltage and pulse-width alignment and a zero-insertion-force socket. A five-foot flat ribbon-cable connects the programmer to the read-only PROM socket via a 24-pin plug. Kit, $249; assembled, tested and aligned, $295. (For a limited time, OAE is shipping the assembled, tested and aligned unit for the kit price.)

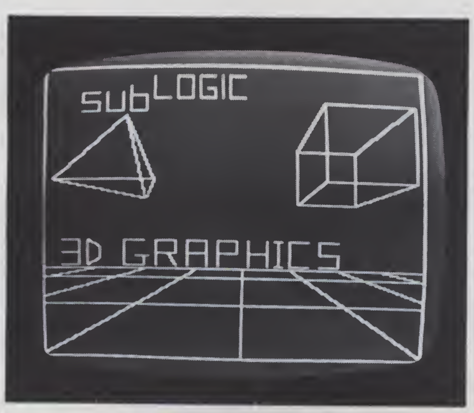
Oliver Audio Engineering, Inc., 676 West Wilson Ave., Glendale CA 91203.

Z-TEL Text Editing Language

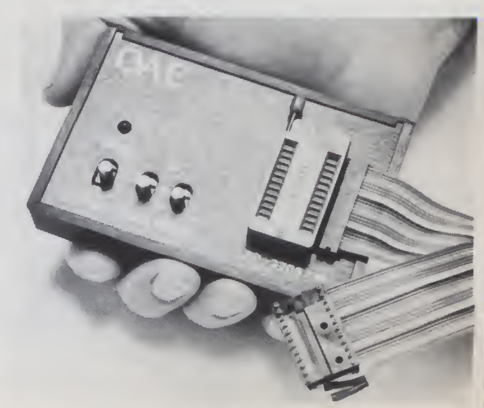
Z-TEL, a Z-80 text-editing language, is a utility program designed to provide a set of techniques for editing and manipulating text files. It can easily move large blocks of text inside the buffer, enabling the user to avoid the deletion and manual retyping of text.

The macro capability of Z-TEL provides the option of tailoring the program to the user's individual needs. A macro expression is a string of commands stored in one or more of Z-TEL's ten text registers. When a macro command is typed, Z-TEL will execute the series of commands using specified text register(s) and eliminate the repeated typing of similar or identical commands.

Z-TEL includes detailed error detection; most error messages



Sublogic micrographics.



OAE PROM programmer.

are displayed in the form of numbers. The manual gives detailed instructions for correction; other error messages are textual and self-explanatory.

Z-TEL provides decision-making capabilities and transfer of control (branching) from one part of a command string to another. Additional features include nested iteration and backward search.

Z-TEL is a relocatable ROM program requiring less than 7K of memory, and it runs on a microcomputer. It is available on paper tape for $50, on cassettes for $40, and will be available on disk.

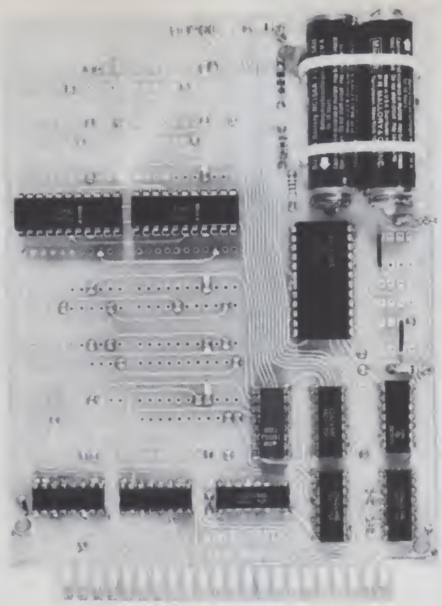
Technical Design Labs, 1101 State Road, Princeton NJ 08540.

Floppy-Disk Interface Card

Data Systems Design, Inc., has a new card for interfacing their DSD 210 floppy-disk system to DEC's LSI-11 minicomputer.

Capabilities of the new interface card (DSD 210-L11A) include a hardware bootstrap, dynamic-memory refresh logic and bus termination circuitry. Because it combines the principal features of DEC's REV-11 card with those of an existing Data Systems interface card, the DSD 210-L11A eliminates the need for the DEC card and saves one Q-bus slot in the LSI-11. It is priced at $319.

Data Systems Design, Inc., 3130 Coronado Drive, Santa Clara CA 95051.



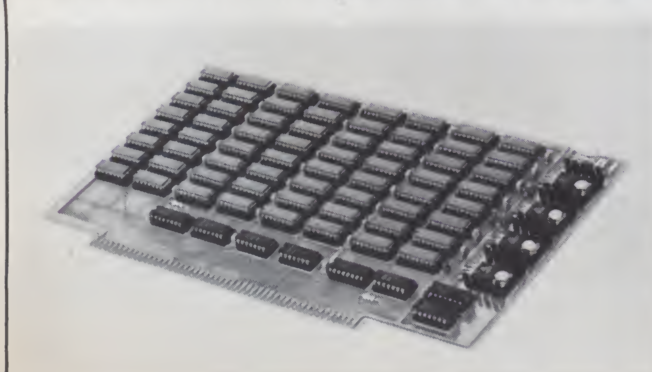
Wince nonvolatile RAM.

Heathkit Model H10.

8KS RAM Board

Associated Electronics' 8KS RAM board is plug compatible with the Altair 8080 and Imsai 8080, or any Altair-bus system. The board uses low-power, 500 ns RAMs with no wait cycles required. Other features include low-power Schottky support chips and DIP switch selection of memory address assignment and memory protect. Low profile sockets are provided for all RAMs and ICs. Edge connector contacts are gold plated. Prices: kit, $189; assembled, $299.

Associated Electronics, 1885 W. Commonwealth, Unit G, Fullerton CA 92633.



Associated Electronics' 8KS RAM.

Nonvolatile CMOS RAM

Microprocessor memory expansion with nonvolatile memory is now possible with the Wince CMOS RAM/battery module. Memory is retained during power-off conditions including when the module is unplugged from the system bus. Two size AA nickel cadmium batteries allow for power-off periods up to one year. The module can accommodate up to 2K bytes in multiples of 256 and has write protection. The price is $399. Wince Micro Modules are the only 6800-based uP modules available on industry standard 4½ × 6½ inch 44-pin printed circuit boards.

Wintek Corp., 902 N. 9th Street, Lafayette IN 47904.

Audio Engineering Card Cage

Audio Engineering has introduced a card cage for use with Motorola's MEK6800D2 kit and other compatible computer systems. The design incorporates molded plastic card guides, aluminum socket supports and a motherboard. All components are self-aligning for ease of assembly.

The unit is available in 5- and 11-slot versions. Spacing between positions is 1 3/8 inches, allowing wire-wrap sockets to be used at all board locations.

Fully assembled 5-slot cage, $84.95; 11-slot version, $157.50. Unassembled, $69.95 and $137.50, respectively.

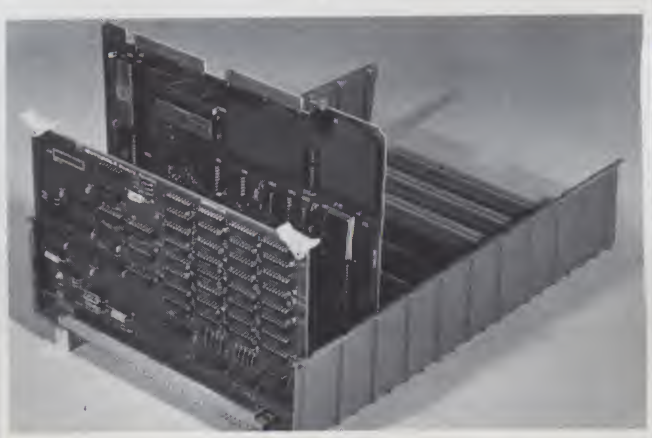
Audio Engineering, 121 Wisconsin N.E., Albuquerque NM 87108.

Heath Paper Tape Reader

Heath Company's H10 Paper Tape Reader/Punch Unit is intended for use with Heath's own computers, the H8 and H11; the H10 is a mass storage peripheral unit that will also function with all other digital computers.

The H10 uses standard 1-inch-wide roll or fan-fold 8-level paper tape. The reader section reads

(continued on page 122)



Audio Engineering card cage.

BOOKS

Microcomputers/Microprocessors: Hardware, Software, and Applications
Hilburn and Julich
Prentice-Hall, 1976, $18.50

This rather expensive book seems to fulfill two approaches. First, it is an excellent reference book for the novice microcomputer user and/or designer. Second, it treats much of the material in a textbook manner: Material is easy to follow and not too technical, and questions are asked at the end of each chapter to reinforce the reader's newly acquired knowledge.

A third, disguised, aspect of this book lies in the reference listings. The magazines and books read like a Who's Who in the computer industry: *Computer, Digital Design, Electronic Design* and *Spectrum,* to name just a few. The microcomputerist will certainly find this list useful during his or her search for supplemental reading material.

Although both authors are members of the electrical engineering department at Louisiana State University, their approach to the material yields a good mixture of hardware (digital logic, interfacing devices, design and architecture methodology, etc.).

For the hardware novice, the chapters dealing with digital logic circuits and I/O interfacing devices will provide a good foundation on which to build up a better working knowledge of microcomputer hardware. Hardware information includes gating structures, Boolean Algebra, ROM, RAM and more.

The software beginner is not left out. There is a chapter on numbering systems, and one dealing with microcomputer programming languages.

However, the software chapter is lacking in some respects. The authors deal lightly with machine and symbolic languages, then enter into a discussion involving Intel's PL/M as run on their MCS-8/80 system. Alas, no BASIC!

There are a few more shortcomings: Since the authors jammed so much fundamental information into this book, in-depth discussion for any particular piece of hardware or software was not possible. It is left up the the reader to use the references for further treatment of lightly discussed topics.

For the examples and illustrations in the software portions of the book, the authors chose the Intel MCS-4/40 or MCS-8/80 system. A comparison of instruction sets for the more familiar microprocessors shows the above systems to be quite representative.

The appendix lists instruction sets for: Intel's 4004, 4040, 8008 and 8080; the National IMP-4 and PACE; Rockwell's PPS-4 and PPS-8; the 6800 from Motorola; and, last, but not least, the RCA COSMAC.

This book is a worthwhile addition to the novice microcomputerist's library. It should also serve as a quick reference guide for the accomplished microcomputerist.

Len Gorney
Clarks Summit PA

The First Book of KIM
Edited by Butterfield, Ockers and Rehnke
O R B, Argonne, IL, $9.00

If you have a KIM and you don't have this book, you're missing out! The introduction states that *The First Book of KIM* is "dedicated to the person who just purchased a KIM-1 and doesn't know what to do with it." The statement is a little misleading in that it implies the book is directed toward, and only has good information for, the beginner. This book has something for everyone! Even if you have had a KIM for a year or two you will probably find some useful programs.

The book has been assembled from contributions by various authors who have written for the *KIM-1 User's Notes* and/or directly for the book. KIM owners seem to be some of the more enthusiastic and imaginative among the ranks of computer hobbyists. Things that have been, and will be, done with this single-board computer (designed for and sold mostly to the industrial market) are staggering in their scope!

The First Book of KIM is divided into six sections. The first deals with programming the KIM from the beginner's standpoint. Aside from fundamental 6502 programming, such techniques as how to display values and reading the keypad using the KIM monitor are covered. This is normally the kind of information that would take some effort and time for most novices to dig out.

The majority of the book contains game programs, and there are more in the second section. There are 28 programs in this section; and what's amazing is that they were all written to run with just *the bare KIM system.* They don't require an ASCII keyboard (the keypad is used) and they don't require a video display (the 7-segment displays are used). Actually, there are a few useful and practical programs, including: a digital stopwatch, cw code practice program, a 24-hour clock program and a binary/hex teaching program, in addition to the games.

The games range from Duel (a reaction-time program for two players) to a complete Wumpus game to a music generation program to just about anything you can think of! The main drawback of the book is that a cassette tape or paper tape of all the programs is not offered. Punching in the various programs from the hex keypad gets tiring in a hurry (but it is, after all, *low-cost* software that we're after).

The third section contains some diagnostic and utility programs for the KIM. Jim Butterfield and Julien Dubé's Hypertape (originally called Supertape) program is described for those who want to load KIM programs in one-third the normal time. Others include a memory test, mini-disassembler, relocation program, sort program, a program that will get the ID and starting address from a tape (in case you forget it), and a unique program for adjusting the cassette interface phase-locked loop circuitry (in this program, no test equipment required!).

Next is a subject near and dear to every KIM owner's heart: expansion. Memory, video displays, keyboards, printers and external mass memory are the hardware items discussed. Expansion software (which is always needed) includes disassemblers, interpreters (BASIC, FOCAL, etc.), text editors and math packages.

The fifth section considers interfaces for the KIM-1. A burglar alarm is used as an example in the discussion. This is a definite aid in helping the reader relate to the hardware and software necessary to interface peripheral equipment.

Further guidelines for generating KIM software, sources of KIM publications and literature, and other miscellaneous information are contained in the final pages.

I hope there will be a *Second Book of KIM.* Perhaps the second version will contain programs using a video display and ASCII keyboard. As much as I'm looking forward to a sequel, I doubt that I'll be through with this one when it arrives!

Gene Christianson
Santa Barbara CA

Scelbi "8080" Software Gourmet Guide and Cookbook
Robert Findley
Scelbi Computer Consulting, Inc., $9.95

If the 8080 assembler on your computer accepts the old 8008 mnemonics, or if you want to really confuse yourself by trying to learn two intermingled instruction sets, then this book is for you. Unfortunately, my Imsai 8080 assembler does not accept 8008 mnemonics, nor do I wish to learn them, so I was disappointed by the book.

For the most part, it is an update of an earlier book on machine-language programming for the 8008 from the same publisher. In many cases, such as in the presentation of parts of the instruction set and sample routines, this is a carbon copy of the earlier edition.

I wanted a book that explained the instruction set and presented some good general-purpose routines, and the description of the *Gourmet Guide and Cookbook* sounded great. The problem with it is best stated by

(continued on page 120)

KB CLUB CALENDAR

Steve Fuller

Boston MA

The Boston Computer Society's meeting coordinator, Jonathan Rotenberg, provides us with an update on his organization.

Meetings are held at 7 pm on the fourth Wednesday of each month (except July) at the Commonwealth School, 151 Commonwealth Ave. Each meeting is divided into three parts consisting of a guest-speaker presentation, an "answers to questions" period, during which the combined knowledge of the Society is available for problem solving and, finally, an open discussion period when members can get acquainted and use one of the available computers. The club is also involved in activities such as forming software exchanges and designing PC boards.

Admission to the first meeting is free; dues are $5 per year. Membership includes admission to all club-sponsored activities and notification of each meeting by mail. The Boston Computer Society may be contacted at 17 Chestnut St., Boston MA 02108. (617) 227-1399.

Las Vegas NV

Cyrus N. Wells is president of the recently formed Southern Nevada Personal Computing Society, Inc. (S.N.P.C.S.). Meeting schedules and club information are available from secretary/treasurer Edna H. Wells, 1405 Lucilee St., Las Vegas NV 89101, or call (702) 642-0212.

Birmingham AL

Several enthusiasts have formed the Birmingham Microprocessor Group and are holding monthly meetings to pursue their interest in computers.

Address your inquiries to the group at PO Box 8072, Birmingham AL 35218, or call club president Jim Anderson at 879-9630.

Syracuse NY

Computer Hobbyists in Processing-Syracuse (CHIPS) extends an invitation to all central New York staters to attend their monthly meetings.

According to vice president John Morrell, the club has been active for about a year. Meetings feature hardware and software demonstrations, and are open to anyone interested in microcomputers.

If you'd like more information, write to CHIPS, c/o J. A. Green, General Electric Company, Court St. Plant #3, Room 16, PO Box 4840, Syracuse NY 13221.

PET User Group

A user group has been formed for people interested in the Commodore PET 2001 computer.

According to Gene Beals, the purpose of the group is to exchange applications, programs and hardware expansion techniques, and to provide user feedback. The first year membership is $5 and will include the User Notes publication.

Details are available from PET User Group, PO Box 371, Montgomeryville PA 18936.

Norfolk VA

The Tidewater Computer Club holds its meetings on the first and third Tuesdays of each month at the Janaf Office Building. Further information is available from Jack Starr, (804) 587-2575.

TRS-80 User Group

Hobbyists wishing to join a group to exchange programs and technical information for Radio Shack's Z-80-based TRS-80 computer should send an SASE to R. Gordon Lloyd, 7554 Southgate Rd., Fayetteville NC 28304.

San Jose CA

If your interests lean toward 6800 micros, the 6800 Computer Club would like to hear from you. Ray Boaz says, "Anything and everything involved with 6800-based microcomputers is of interest." Hardware and software assistance is offered to all system owners and users.

Meetings are held at the University of Santa Clara on the first or second Tuesday of each month. Write to the 6800 Computer Club, Box 18081, San Jose CA, for detailed information.

Lincoln NB

Hubert O. Paulson, Jr. invites hobbyists in the Lincoln area to participate in the monthly (first Wednesday, 7 pm) meetings of the Lincoln Computer Club. The group's current meeting place is the South Branch Library at 27th and South Streets.

Details on club activities are available from Hubert at 422 Dale Drive, Lincoln NB 68510.

Bristol RI

The Rhode Island Computer Hobbyists have announced a tentative meeting schedule of the third Tuesday each month from January through May. For schedule confirmation and club details write RICH, PO Box 599, Bristol RI 02809.

Portland OR

The Portland Computer Society, 3763 S.E. Division St., Portland OR 97202, meets on the third Saturday of each month. Meetings are held at 1 pm in the Pine Room, Sylvania Campus, Portland Community College. Contact club secretary Percy G. Wood for information and a copy of the club newsletter.

SOL Newsletter

A SOL user group newsletter, SOLUS News, is available from editor Stan Sokolow, 1690 Woodside Road #219, Redwood City CA 94061. A $4 donation is requested to cover printing and mailing costs.

National Park NJ

The National Park Municipal Building, 7 South Grove Ave., is the meeting place of the Computer Amateurs of South Jersey. Charles Knott tells us that user groups have been formed for the 6800, 6502 and 8080 microprocessors, and welcomes all novice and professional computerists to join the group on the last Tuesday of each month at 7:30 pm.

You may correspond with the club by writing c/o Charles Knott Associates, 3088 Congress Road, Camden NJ 08104. (609) 541-1010.

Las Cruces NM

James Leatham sends word of "yet another computer club." The Las Cruces Computer Club (LC^3) meets on the first Thursday of the month at the New Mexico State University at Las Cruces (Walden Hall, Room 229, 7:30 pm). Discussions include hardware, software and microcomputer applications topics. A $1 admission fee is charged to cover club expenses.

Write to LC^3, c/o Leonard Fetterhoff, 2085 Payne Court, Las Cruces NM 88001, or call him at 522-3381 or 678-2361.

Rockford IL

Computer hobbyists in the Rockford area now have a place to go on the second Wednesday of the month. The Blackhawk Bit-Burners Computer Club invites you to share questions, experiences and frustrations with their membership.

Write to club president Bill McKenzie, PO Box 5411, Rockford IL 61101.

This column is available for you to report on your club's activities such as regular meeting schedules, special events or programs, hardware/software demonstrations for club members or the general public, or any endeavor that will interest your fellow computerists. In short, we hope it will benefit your club and hobby computing.

Send news of your organization's activities to:

KB Club Calendar
c/o Steve Fuller
PO Box 218
Spofford NH 03462

LETTERS

Some Comments: SC/MP-II Retrofit

I have just run across some information concerning the SC/MP-II and the SC/MP kit that might be of interest to your readers: National Semiconductor has released what it calls a retrofit kit for the $99 P-channel SC/MP kit. This kit costs about $20 and consists of an N-channel SC/MP-II, some miscellaneous components and a 2 MHz crystal that replaces the 1 MHz crystal in the original kit.

Care should be taken in reading National's promotional literature on the SC/MP-II and the retrofit kit, as some confusion may arise. The SC/MP-II is promoted as being faster than the P-channel version (it can operate with a 4 MHz clock) and as having software compatability with the P-channel version, but requiring a single +5 volt supply, instead of the +5, −7 volts required on the earlier version.

What is not made apparent in all of this is that the SC/MP retrofit kit, with its 2 MHz crystal, will not cause your SC/MP kit to run any faster overall than it did before with the original 1 MHz crystal! This stems from the fact that the SC/MP-II takes four clock cycles to perform a processor microcycle, while the original SC/MP and the SC/MP-II took only two. The number of microcycles necessary for the SC/MP and the SC/MP-II to perform each instruction in the set is almost identical, with but a few differences in favor of the SC/MP-II.

The net result is that for the same clock frequency the SC/MP-II will operate at about half the speed of the older version of the SC/MP; the SC/MP-II is really only faster when driven at clock frequencies above 2 MHz.

The SC/MP retrofit kit is, then, a retrofit in the classic sense: a kit that allows a new component to be installed in an older piece of equipment, with little or no change in the overall performance of the circuit. After modification, the SC/MP-II kit will operate at the same rate as the old kit, and will still require +5, −12 volts to support its memory. Although the software is compatible, it should be noted that some program timing changes may be necessary, as well as changes in programs using the SC/MP's interrupt enable (ENIN), which is changed in sense on the SC/MP-II.

Those readers considering the purchase of the SC/MP retrofit kit for the purpose of upgrading the performance of their present SC/MP kit should consider these factors before laying down their money.

Don Taylor
Route 2 Box 520
Poulsbo WA 98370

Thanks for letting me respond to the letter from Don Taylor. I refer your readers to Compute *(National Semiconductor's microprocessor newsletter) Vol. 3, #5, pg. 19. Note that the SC/MP-II Retrofit was initially proposed for evaluation purposes. To keep the cost down, a ROM was not included in the kit, but the changes in the Kit Bug program refect the higher speeds and were published alongside the announcement in* Compute.

The polarity of some of the signals was changed from SC/MP-I to SC/MP-II. ENIN was one of them, but this does not change the way programs are written for SC/MP.

Georgia Marszalek
Microprocessor User Group
National Semiconductor Corp.

Imsai Mods Revisited

I was glad to see that Don Walters noted the power-supply problem in his construction article, "Put Your Imsai on the Rack!" in the October issue. This is not a new problem—Micro-Circuits Hawaii has built approximately 30 of these units and has notified Imsai of the details. Still, the documentation has not been changed.

The one argument I have with this article is that we at MCH have experienced heat problems when too many sockets are placed on the motherboard. We recommend to all our customers that sockets be placed in every other space to prevent this.

As a servicing aid, our Imsai is also placed in a rack, but on rollers with the top permanently bolted to the rack so that the unit can be pulled out to expose its "innards." One-half inch less width would have allowed side mounting of the rollers—virtually impossible at the given width.

I am impressed that someone else is finally using an air filter. It seems that Imsai would have come up with that one first.

The modification Don made at U13 of the CP-A board has not occurred in our experience. I suggest, possibly, an out-of-tolerance 74107 (U18).

Thanks for a beautiful magazine.

Micro-Circuits Hawaii
Honolulu, HI

The Fairchild Video Game

I'm just getting started in small computers and have a small budget to work with. Your magazine seems to be the best for someone like me.

I noticed in your August issue that someone asked about the Fairchild game and whether anyone was using it for other than games applications. I'm trying, but I need more knowledge. Has anyone sent in anything about the Fairchild game with F-8?

I also have an ATC (Atlantic Technology Corp) 2000 Data Display Terminal and keyboard. The keyboard is not ASCII encoded but seems to be made for the terminal.

The terminal doesn't come on and no filament voltage is indicated.

Can you please tell me where I might get information on this terminal and keyboard?

Dennis W. Choinski
2366A S. Howell
Milwaukee WI 53207

Quite a while ago, I received a very good article by Jim Huff on dissecting the Fairchild video game. Shortly thereafter, I was in contact with the folks at Fairchild, and they told me they were working on an 8080 interface for the game . . . and would write an article on it. I held off running Jim's article in anticipation of this one. Since it's been almost a year, I guess Fairchild is not going to come through. You'll be seeing Jim's article in the future.—John

Marriage: Morrow Front Panel—Poly Video

The following concerns a problem I recently encountered while troubleshooting a friend's PolyMorphic video board on my KLUGE-8. The "standard" S-100 bus bites again! When a PolyMorphic video board is plugged into a bus driven by George Morrow's CPU/Front Panel board, strange things happen. Synch seems unattainable. Occasionally, a pattern of dozens of tiny screens will momentarily lock in, only to tear into garbage again.

The solution turns out to be simple. The Poly board expects a 2 MHz clock (inverted Ø2) on pin 49 of the bus. George, apparently deciding that three 2 MHz clocks on the bus were more than enough, provides a buffered 18 MHz clock at that location. The fix is easiest to do on the Poly board as follows.

Cut the bridge at JMP5, as instructed, to implement the onboard crystal clock.

Jump the lower right hole at JMP5 to bus pin 24 (Ø2). Be careful to keep the solder near the top of the finger, not down on the connecting area.

The board will now run with the Morrow CPU.

James C. Matthews
Montgomery AL

We're interested in any other reader fixes to problems of S-100 ≠ S-100.—Wayne.

Six Weeks? Why?

As promised, I received my first copy of *Kilobaud* six weeks after I sent in my check—which brings to mind a question I've often asked about magazines. Why is it I can write a busy friend and get a reply in two weeks; but, when I send in money for a subscription, it takes so long for the return of that month's issue (which I never get unless I pay extra for it)? I realize the company has to be sure my check won't bounce—but what else happens in the six weeks?

I think the question is particularly apropos of *Kilobaud* since it seems to be one of your intentions, or contentions, that microcomputers are useful. Couldn't you assign one person to one minicomputer for about ten minutes each day to add the scores of new subscribers to the list and then (if that person isn't tired yet) write or call the different banks in question to verify funds? If computers are so fast, it seems maybe a future for them could be found in processing subscriptions.

Michael Windolph
Chaska MN

Sometimes it doesn't take six weeks. If, for example, you consider that it takes the post office approximately two weeks to deliver your copy once it has been mailed . . . plus another week for the printer to get the labels from the computer printout and then schedule them to be put on the wrappers . . . plus the delay between the time the subscription is sent to the magazine . . . handled by the circulation department . . . sent to the computer data-input firm . . . and then put into the computer at the once a month update . . . you get the picture.

We tried doing all this here and were unable to get the computer to handle it; so we had to go back to an outside computing firm. When our programs are working better, we'll try it in-house again. This (we hope) will speed things up a bit.—Wayne.

How About a "Hyper" ACR?

Jim Butterfield's article about Hypertape in the November issue inspired me to try an experiment with a Mits 88-ACR cassette interface board. Like the KIM, the 88-ACR uses a PPL. At the recommended 300 baud, the ACR uses slightly over six cycles of lower-frequency tone. It followed that it might work at a higher baud rate.

I wired a DPDT switch to change the appropriate pins to select between 300 and 1200 baud. The initial test bombed. I had to make a 1200 baud tape of 167 octal. By single-stepping through an IN data port so that the input appears on the data lights, I was able to realign the ACR receiver section for proper reception. Now it works fine.

I found that the programs supplied with the ACR were in error. The ACR would work—sometimes. I solved the problem by inserting an IN data port to clear the UART before proceeding to the status port sampling routine.

I modified the bootstrap routine to load incrementing. When I record a tape I dump the whole page containing the tape load and dump routines first. It was necessary to insert a delay by counting a register up or down 256 (decimal) to allow time for the loader routine to get set up.

Do you have any information on the Tarbell Disk interface? Can it be used with the Shugart minifloppy?

Joe McCarty
Amarillo TX

We have an article in this issue on the Tarbell Floppy Interface, Joe. And, yes, it can be used with a Shugart.—John.

Dumper Loader Block for Mits 88-ACR
(no checksum or overlay)

Bootstrap PCHL's to HHH 000	LXIH	Start address of load
	LXIB	End address of load
	IN	Data port of ACR
STAT	IN	Status port of ACR
	RRC	
	JC	to STAT
STAT1	IN	Status port of ACR
	RRC	
	JC	to STAT 1
	IN	Data port of ACR
	MOVMA	Store data
	MOVAL	
	CMPC	
	JNC	to INX
	MOVAH	
	CMPB	
END	JZ	infinite loop END
INX	INXH	
	JMP	STAT1
DUMPER . . . . Start	LXIH	Loader Dumper block start address
	LXIB	End of block address 377
STAT2	IN	Status port of ACR
	RCL	
	JZ	to STAT2
	MOVAM	byte to output
	OUT	Data port of ACR
	MOVAL	
	CMPC	
	JNZ	to INX1
	MOVAH	
	CMPB	
LOOP	JZ	to NEXT first time INF loop 2nd time
INX1	INXH	
	JMP	to STAT2
NEXT	LXIH	data for INF LOOP at LOOP
	SHLD	modify LOOP arguments
	LXIH	Start address of data to dump
	LXIB	Stop address of dump
	MVID	Zero D
TIME	DCRD	
	JNZ	to TIME
	JMP	to STAT 2

Mits 88-ACR Bootstrap Modified

	000	LXI H	With the page where the loader is on tape
	003	IN	The data port of the ACR to clear the UART
TAG	005	LXISP	022 000
	010	IN	The ACR status port
	012	RRC	Move bit 0 into the Carry flag
	013	RC	Return if the carry is 1 to TAG
	014	IN	The ACR data port
	016	MOVMA	Store the data in memory
	017	INRL	Increment the memory pointer
	020	RNZ	If L not equal to zero return to TAG
	021	PCHL	Go to the loader block
STACK	022	005	Preloaded stack the return to TAG
	023	000	

Lifetime Update

Terrence Lukas's article, "Lifetime Program," in the November *Kilobaud* is very timely. It drives home the importance of our bad habits, particularly smoking and being overweight. It would probably be useful if suggestions were made at the end of the program on how to improve your longevity—getting more exercise, for example.

Using my actuarial training to review the program for technical faults, I could not find any serious things wrong. I couldn't identify the mortality table from which the life expectancies were determined; I would recommend the U.S. 69-71 Life Table. If anyone needs a copy of the life-expectancy tables, they can write, enclosing an SASE, and I will send it.

Also the tests do not ask some important questions that a life-insurance underwriter would ask. They are:

1. Have you bought a medically underwritten life-insurance policy at standard rates in the last three years? (+3) The reason is that people who have recently purchased a life-insurance policy normally have very low mortality rates for the next ten years.
2. Do you have medically diagnosed heart disease? (−15)
3. Do you have hypertension? (−7)
4. Are you diabetic? (−9)
5. Do you have emphysema or chronic bronchial problems? (−4)

One other point is that most people do not understand what life expectancy means. It is some point in the future when one-half the people age X will be dead. It is not how long *you* will live, but the average *everyone* aged X today will live. I hope your readers find this information helpful.

Robert C. Fruit
Computer Math Notebook
100 Fuller Road
Hinsdale IL 60521

Son-in-Law of Submarine: Modified

With mild nostalgia, I noticed the Smith and Marzano article, "Son of Submarine" (*Kilobaud* No. 11). My original SR-52 has been mainly relegated to the role of super calculator, since I've outgrown it in favor of a micro. I still use some of my existing design programs, but original programming and game playing has fallen by the wayside. I thus had a lump in my throat as I pressed the LRN key and began entering the program. This soon changed to jubilation, and my cohorts cheered me on as I chased the little beggar around the Indiana [sic] Ocean.

After playing a few rounds, I noticed a couple of improvements that could be made. One involves slight rearrangement of the subroutines and branches, and elimination of some duplicate parenthesis and FIX statements in the "random" subroutine. These modifications afford slight economies in running time.

The other change concerns score keeping. In this game, it is not a matter of whether one hits the submarine or not, but how long it takes to do it. It is a foregone conclusion that eventually the sub will be hit. The challenge comes from seeing how fast the sub can be caught. Thus, some form of scoring is advisable.

When the sub has been hit the display shows a flashing 9.99E99. If it would flash the number of depth charges expended in chasing the sub, the ship's captain would not only know that the sub had been destroyed, but would also know how long it took to do it. If it took ten shots or less, the captain could pat himself on the back, and if over 25, he could be thankful for the trailer full of depth charges being towed behind the ship.

The following listing accomplishes these changes while actually using less program space. Happy hunting!

Son-in-Law of Submarine Program Listing.
(Comments remain the same as the November article.)

Location	Key Strokes
000	LBL' 1'
002	5 +/−
004*	+ .1 x
008	LBL' 8'
010*	100 x (7 Y^x 9
018	x RCL 03
022*	x 5 +/− INV LOG'
027	− (RCL − .5)
034*	FIX' 0 DMS')
038	STO 03 =
042*	DMS' FIX' 2
045	RTN'
046	LBL' A
048	STO 05
051	HLT
052	LBL' B
054	STO 06
057*	1 SUM 08
061	RCL 06
064	− RCL 02
068*	= X^2
070*	+ (RCL 05 − RCL 01)
080	X^2 =
082	x
083	STO 07
086*	if zro' =
088	− 5 =
091	if pos' +
093	INV DSZ' C
096	LBL' +
098	SBR 1'
100	SUM 01
103	SBR 1'
105	SUM 02
108	RCL 06
111	− RCL 02
115*	= x^2
117	+ (RCL 05 − RCL 01)
127*	x^2 = x
130	STO 04
133*	HLT
134	LBL' =
136*	RCL 08
139*	GTO if err'
141	HLT
142	LBL' C
144*	CLR
145*	0 STO 08
149	SBR 8'
151	STO 01
154	SBR 8'
156	STO 02
159	5 STO 00
163	HLT

**New additions or changes.*

Philip T. Hodge P.E.
Schererville IN

Remember that every time you sink a submarine you are creating lots of widows and orphans—so take it easy with this type of thing.—Wayne.

First Letter

I've been an electronics technician for 16 years, and I've read them all and written to none, but your mag and staff are unreal. How can you continue to keep up with yourselves in quality and quantity? Each issue is better than the last. I don't know how you do it, but please keep up the good work.

I believe you have the only uP publication that always touches base with everyone—novice to PhD—every issue. I don't want to miss a single page.

PS. My vote (as a KIM-1 owner) is to keep Rick Simpson. Once again—thanks for all the goodies in *Kilobaud.*

Larry Pearson
Warminster PA

NVM Memory?

Thank you for the great November issue—the best yet. The magnetic bubble memory article was fantastic; now let's see one on the McDonnell Douglas nitride oxide silicon NVM (Nitron) memory chips.

I am in the process of building a Byt-8 and would like to correspond with anyone having a similar system. What problems have you had getting the system up and running? Also, with this system, do I need an active termination; and if so, what would be the design of such a termination strip?

On the interfacing of an alarm system to a computer—I might be able to help answer some questions, as I have had some experience working with alarms. If anyone has any specific questions, enclose an SASE, and I will be more than happy to help in any way possible.

I am looking forward to more articles on interfaces, assemblers

and anything else involving systems design, as I am trying to build my own additional S-100 boards for memory and I/O.

David C. Riness
1549 Hillside Dr.
Glendale CA 91208

The Latest in Disk Systems

Our company is a manufacturing and sales outlet for a new generation of micro and mini computers. We have a franchise aim for a hobby computer store with a strong emphasis on the small-business market.

During "Ma Bell's" yellow-page campaign, our company produced two ads showing the internal structure of a business-retail division. As these two ads were given the same address for billing, the printers combined the ads and came up with a yellow-page ad announcing our company as the proud (?) producer of the only *Sloppy Disks* in America that are double-density formatted.

We figured that such an engineering feat as *Sloppy Disks* would be of interest to your public. Anyone interested in our design structure will be sent an information package.

Douglass C. Boseman
ComputerCo, Inc.
Charleston Hgts SC 29405

PUBLISHER'S REMARKS

(from page 4)

possible. It helps if you can recount it to someone, but an assistant isn't absolutely necessary.

When you reach the end of the material go right back and go through it again, covering every single detail. You'll find that some of what you missed the first time through will come to mind this time. About the sixth time you go through the material you'll find you are able to run it like a speeded-up movie . . . the whole article can be skimmed in a few minutes. If you do have a helper, you can have him or her stop you randomly as you scan, and you'll find that the part of the article you are scanning will spring out in full detail.

You can check the state of your memory every few weeks and run a quick refresh. A couple of scans and you should be right back up to speed, running an entire book in minutes. Yes, the material is all there, and you can search it easily by any cross-reference you care to use . . . or you can stop at the snap of a finger and give complete details.

Now you know about the narrow columns in *Kilobaud.*

EDITOR'S REMARKS

(from page 5)

same old complaint about the PET's keyboard and the lack of strings with the TRS-80 prevailed . . . but the enthusiasm for both systems overshadowed any criticism.

That's just some of what was going on in one day in good ol' Lompoc CA.

Radio Shack Happenings

I just borrowed a Radio Shack TRS-80 system for about four days . . . and I even got to use it some! My family took it over

most of the time, and I was left with the late-night and early-morning hours for playing and experimenting. There's a compelling explanation for why the thing experienced so much use around here (it's not as though my household was seeing a computer for the first time!): Some *very* interesting software packages were provided with the machine.

My nine-year-old has been having trouble with his math, and the 3-cassette package, "Education—Math 1," was a spectacular aid. (I probably should have taken him out of school for a couple of days and left him with the computer!) I'm still not convinced my wife was entirely sold on the "Kitchen" and "Personal Finance" packages, but it's only because they're the sort of thing you have to use . . . not just demonstrate. Blackjack and Backgammon had the machine tied up for hours. My kids have played blackjack on a computer before—but never with the graphics of a TRS-80.

The machine was loaned to me by Tom Preato, a friend who owns a Radio Shack store in Oxnard CA. Tom is a smart cookie. When the TRS-80 was first announced, he knew it was going to be a hot item . . . so he ordered one for himself. As you probably know, Radio Shack has a policy of providing a demo system to a store *after* they have sold three systems (kind of a "Catch-22").

Tom says buying his system is one of the best investments he ever made. As a result of having a TRS-80 system in the store, he's been selling a system every two or three weeks, on the average. He's also planning to use the system in the store for inventory and accounts receivable and payable (after expanding to Level II BASIC, a floppy and 16K of memory). Then he'll *really* have something to show people!

You'll notice that I haven't mentioned anything about the hardware. Actually, I didn't even open the thing up to look inside . . . I was so intrigued by the software. And . . . software is going to be the key for *any* personal system arriving on the scene today. All indications are that Tandy Computers will be a force to reckon with.

Tandy is hiring programmers (and engineers) like crazy! This programming staff will be put to work cranking out business-applications programs for *all* of the computers carried by Tandy Computer Stores. The list is impressive; just take a look at a Tandy ad and you'll see what I mean. In addition to the popular 8080 and 6800 systems, they'll also be carrying Computer Automation's "Naked Mini" and Alpha Microsystem's AM-100 (both multi-user systems). Their engineering group will be kept busy interfacing custom systems, providing service and maintenance and most definitely, coming up with new designs.

Photo 2. Don't be misled by Bill Sherrill's thoughtful look. He's interested . . . but not in buying. He's a dyed-in-the-wool SOL owner.

Photo 3. As usual, the kids continue to tie up computers and deprive us grownups of our fun. (This little guy sure was enjoying himself!)

The new Level II BASIC will cost approximately $100 and be installed (Level I upgrade) at their service centers; turnaround time will be 48 hours. I must admit that I'm a little disappointed with the string capabilities of Level I . . . and I'll be looking forward to Level II. By the way, Level II is a product of Microsoft, according to Bill Gates, president of Microsoft. I think that is absolutely great! Do you realize how may personal computers are now running around with Microsoft BASIC? There's the Altair (both 8080 and 6800), OSI, PET, KIM, Apple (floating-point version) . . . and now the TRS-80. Boy, oh boy! We may be on the road to compatibility! (For your miscellaneous file: the 6800 and 6502 Microsoft BASICs are essentially the same as the 8K 8080 version. There are three enhancements in the 6502 and TRS-80 version. These include long textual error messages, integer variables and nine-digit accuracy.)

Some interesting software will closely follow Microsoft's Level II BASIC. They're developing General Ledger, Accounts Payable, Accounts Receivable, Inventory Control and an extended Payroll packages (I found their Level I Payroll package for 15 employees quite impressive). All of this in addition to even more educational, entertainment and home application programs. (Oh, yes—while we're on the subject of software, I should also mention the Z-80 Assembler/Editor that will be along shortly.)

What about that hardware I almost overlooked? Well, Tandy is shooting for an under-$500 floppy, a low-cost hobbyist printer and an under-$1500 dot-matrix printer for business applications. Beyond a doubt, though, I think the most significant hardware development to come from Tandy Computers is their RS-40 (Radio Shack 40-pin bus) to S-100 interface. Naturally, they want to sell as much in-house memory and other options as possible . . . but look at the versatility they're offering their customers! My gosh, talk about being able to shop around for the best value . . . look at all the S-100 memory boards around! (I'm sure Tandy must have realized that one of the first things to pop up from the hobbyist community would be an S-100 interface, and they just beat us to the punch!)

LEGAL/BUSINESS FORUM

(from page 7)

assembled products. The majority of manufacturers provided 90-day warranties.

More than half the manufacturers believed that their warranties were equal to the industry average. A smaller percentage thought their warranties were better than the industry average, but these warranties were not actually full.

Turnaround times reported by manufacturers ranged from one day to 30 days. It is interesting to compare this figure with the retailers' report on manufacturers' turnaround time. The retailers indicated that manufacturers took three to eight weeks to make repairs. Of course, I did receive fewer responses from retailers, so the odds are that they had not necessarily dealt with these specific manufacturers.

All the manufacturers indicated that there were no hourly labor charges for in-warranty repairs. The rates for out-of-warranty repairs varied from $10 to $25 an hour; the average was $17 per hour. The manufacturers also uniformly reported that the

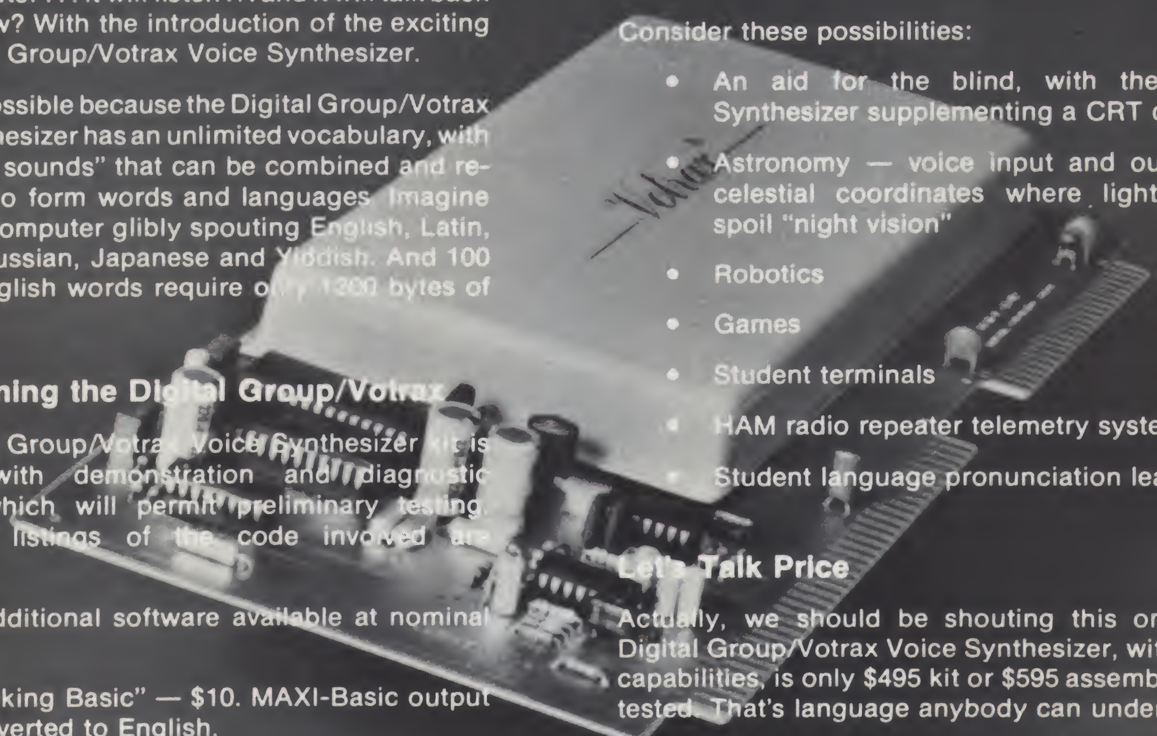
(continued on page 120)

Talk to our Computer... and it will talk back!

(Plainly speaking, it's only from the Digital Group.)

Now, your Digital Group computer becomes more than a silent partner. You can vocally command your computer . . . it will listen . . . and it will talk back to you. How? With the introduction of the exciting new Digital Group/Votrax Voice Synthesizer.

All this is possible because the Digital Group/Votrax Voice Synthesizer has an unlimited vocabulary, with 64 "human sounds" that can be combined and re-combined to form words and languages. Imagine your own computer glibly spouting English, Latin, Spanish, Russian, Japanese and Yiddish. And 100 average English words require only 1200 bytes of memory!



Programming the Digital Group/Votrax

The Digital Group/Votrax Voice Synthesizer kit is supplied with demonstration and diagnostic software which will permit preliminary testing. Assembler listings of the code involved are included.

We have additional software available at nominal cost:

- "Talking Basic" — $10. MAXI-Basic output converted to English.
- "Talking CW" — $10. For impressing your HAM buddies. Requires the forthcoming HAM interface card.
- "Latin and Spanish Talking" — $10. Hear the computer repeat letters and words typed in Latin or Spanish.
- Demonstration Tape — $5. A sample of audio tape and a complete explanation of the system.

Bonus: A basic input circuit is included that may be programmed to understand a small vocabulary of voice commands.

Unlimited Applications

Consider these possibilities:

- An aid for the blind, with the Voice Synthesizer supplementing a CRT display
- Astronomy — voice input and output of celestial coordinates where light would spoil "night vision"
- Robotics
- Games
- Student terminals
- HAM radio repeater telemetry systems
- Student language pronunciation learning

Let's Talk Price

Actually, we should be shouting this one. The Digital Group/Votrax Voice Synthesizer, with all its capabilities, is only $495 kit or $595 assembled and tested. That's language anybody can understand.

O.K., you've listened briefly to what we have to say about the new Digital Group/Votrax Voice Synthesizer. But we *can keep right* on *talking*! Write or call today for *all* the details — music to your ears.

the digital group inc

po box 6528 denver, colorado 80206 (303) 777-7133

How Are You Feeling Today?

biorhythms with your KIM

Larry Woods
Processor Applications

Biorhythm theory states that we have hourly, half day, daily, monthly, seasonal and annual cycles that are thought to somehow trigger and coordinate many of our automatic body functions. If the theory is correct, then by keeping track of the various cycles it should be possible to predict, to a certain extent, the body's potential for performance during the various cycles.

Of the many cycles, the ones most frequently charted and correlated are the physical (23 days), emotional (28 days) and intellectual (33 days). Although the majority of the work being done is empirical and after the fact, it is still extremely interesting.

As shown in Fig. 1, these three cycles are represented as sine waves that start on the day of birth. As each cycle crosses the baseline, it is denoted as a *switchpoint* day. If either or both of the other cycles cross at the same time, rising or falling, it is denoted as a *critical* day. By correlation of switchpoint and critical days, accidents-and-illness researchers have found that people tend to have more accidents or become ill on those days.

A further postulation of the theory states that when the cycles are above the baseline (positive) you're having a *high* for that rhythm. Obviously, you cannot have something for nothing; the theory goes on to say that when the cycle is below the baseline, you are having a *low* for that rhythm.

One interesting relationship of the cycles is that in 21,252 days from your birth date, the cycles will start repeating.

For more information on the subject of biorhythms, check your local library or bookstores. The bookstores in my area had three or four paperbacks by different authors.

The program listing is divided into four parts:

1. $0\text{-}12_{hex}$ contains the required data and temporary storage locations.
2. $20\text{-}8C_{hex}$ contains the look-up tables.
3. $200\text{-}2C2_{hex}$ contains all of the subroutines.
4. $2C3\text{-}3D5_{hex}$ is the main program.

The program starts by converting the total years that you have lived into days. Next, it does the days for the first month, and for each month in the first year. It finishes by doing the days for each month in the last year and for the last month. Then it does the three biorhythm period calculations and jumps to the display routine.

Using the KIM-1 system to display biorhythm graphs does present one small problem: A 7-segment display graph tends to make little sense. Therefore, I converted the three biorhythm periods to a numerical representation with the use of the formula in Example 1 and added a fudge factor when needed. If a cycle was within one day of either of its peaks, I added or subtracted one so that I

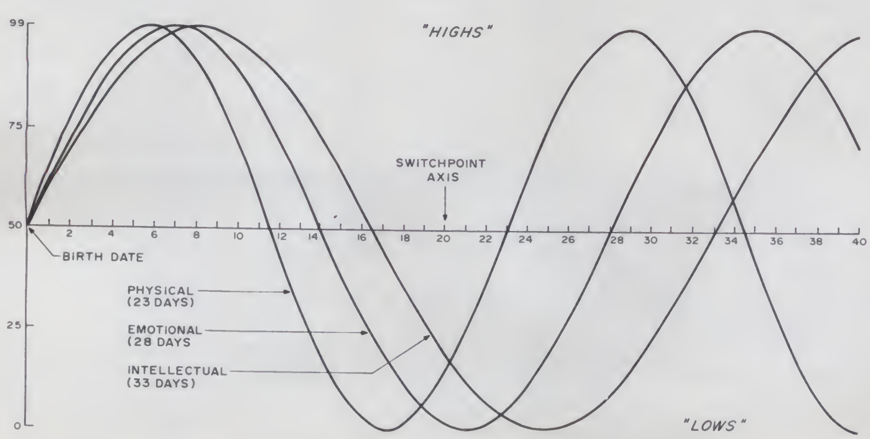


Fig. 1. Graphic representation of biorhythm cycles.

$$\left\{\left[\mathrm{SIN}\left(\frac{360}{\text{total period}} \times \begin{matrix}\text{day}\\ \text{in}\\ \text{period}\end{matrix}\right)\right] \times 100\right\} +50$$

Example 1.

could see the difference between those days and the peak day. Since the sine function ranges from +1 to -1, the display should read between 0 and 100. So, I applied one more fudge factor — the display ranges between 0 and 99. In this manner, all three of the values can be displayed at the same time. The order of display from left to right is: physical, emotional and intellectual. The magnitude

SYMBOL	VALUE	LINE DEFINED	CROSS-REFERENCES									
ADDAYS	023D	105	124	126	128	130	235	249	272	278	294	318
			324									
ADD1	0253	117	109	113								
BDAY	0001	4	271	277								
BLYR	0003	6	212	257								
BMON	0000	3	259	266								
BMYR	0002	5	215	255								
BRPER	03A3	330	273									
CHECK	02A0	171	168									
CONVD	1F48	22	156	159								
CORDEC	0269	134	240	260	287	308	333	340	347			
COR1	0276	143	134									
COR2	026F	138	140									
DAC0	030D	243	247	251								
DAC1	0320	255	220	239	244							
DAC10	0348	274	264	267								
DAC11	0363	290	295									
DAC12	0370	299	292									
DAC13	039E	323	300	316								
DAC14	0391	314	319									
DAC15	0359	286	283									
DAC18	035E	288	285									
DAC3	02F1	230	227	237								
DAC4	0302	238	225									
DAC5	02F3	231	229	233								
DISP	0277	147	351									
DIS1	00FB	23	336									
DIS2	00FA	24	343									
DIS3	00F9	25	151	157	350							
DIS4	0280	151	161									
DIS5	027C	149	162									
EMO	0050	47	342									
HTDAY	0009	12	65	111	112	203						
HTDAYT	0011	20	66	77	93	95						
INCYR	0255	123	231	245								
INT	006C	52	349									
LPYR	0297	167	258	306								
LPYR1	029C	169	167									
LPYR2	02B4	181	172	174	176	178						
LPYR3	02B6	182	170	180								
LYCORR	02B7	187	261	288	309	312						
LYC1	02C1	194	189									
LYFLG	000D	16	181	188	205	302						
MOTBLE	0020	34	274	293	317							
PADD	1741	26	148									
PDAY	0005	8	270	323								
PERCAL	020D	74	332	339	346							
PER1	0221	87	79	82								
PER2	0210	76	91	96	101							
PHY	0039	43	335									
PLYR	0007	10	211	305								
PMON	0004	7	265	286	307							
PMYR	0006	9	214	303								
START	02C3	199	****									
SVTDAY	0200	63	330	337	344							
TEMP1	000E	17	75	81	89	218	263	282	299	310	315	
TEMP2	000F	18	169	256	289	291	304					
TEMPLYR	000C	15	213	217	226	238	241	243				
TMPMYR	000B	14	216	224	236							
TTDAY	000A	13	67	115	116	204						
TTDAYT	0012	21	68	78	98	100						
UTDAY	0008	11	63	107	108	202						
UTDAYT	0010	19	64	80	83	88	90					

Symbol table.

of the display indicates your body's potential for each rhythm, with 0 being the lowest, 43 to 57 being the switchpoint days and 99 being the highest.

To use the program, you need to enter into locations 0 through 7 the month, day and year of your birth (01-15-1943), then the month, day and year (12-30-1977) that you wish to calculate. Next, enter the starting address (02C3) and GO. If the program loaded correctly, the display will show 7639 15.

Since the KIM-1 display routine displays the "data" located at the "address," I had to write my own display routine. Therefore, the only active keys, after executing the program, are ST and RS.

I tried to modify the data used for calculations or the results of any calculations as little as possible, making it easier to single-step through programs.

I chose to do all the calculations in decimal because of the decimal mode in the 6502 CPU. Now I'm not convinced that it was the easiest way, but since all the input data was in decimal and I wanted decimal results, it seemed the best at the time. Also, it did make it easier to debug my calculation algorithms.

One last thought — in locations A, 9 and 8 are the total number of days that you have lived (8 = least significant). I only mention it for the younger ones among us. ■

Program listing.

```
LINE # LOC   CODE    LINE      10          20              30              40

  1    0000          * = $0

  3    0000          BMON      *=*+1     BIRTH MONTH
  4    0001          BDAY      *=*+1     BIRTH DAY
  5    0002          BMYR      *=*+1     BIRTH MOST SIG. YEARS ( 19 )77
  6    0003          BLYR      *=*+1     BIRTH LEAST SIG. YEARS 19(77)
  7    0004          PMON      *=*+1     PROJECTED MONTH
  8    0005          PDAY      *=*+1     DAY
  9    0006          PMYR      *=*+1     MS YEARS
 10    0007          PLYR      *=*+1     LS YEARS
 11    0008          UTDAY     *=*+1     1&10 DAYS TEMP
 12    0009          HTDAY     *=*+1     100&1000 DAYS TEMP
 13    000A          TTDAY     *=*+1     100K & 10K DAYS
 14    000B          TMPMYR    *=*+1     TEMP MSD FOR YEARS
 15    000C          TMPLYR    *=*+1     TEMP LSD FOR YEARS
 16    000D          LYFLG     *=*+1     LEAP YEAR FLAG
 17    000E          TEMP1     *=*+1     TEMP STORAGE
 18    000F          TEMP2     *=*+1
 19    0010          UTDAYT    *=*+1     TOTAL DAYS TEMPS FOR PERIOD CALC.
 20    0011          HTDAYT    *=*+1
 21    0012          TTDAYT    *=*+1
 22    0013          CONVD =   $1F48               KIM CONVERT HEX TO 7-SEG DISPLAY
 23    0013          DIS1 =   $FB                  KIM 7-SEG MSA
 24    0013          DIS2 =   $FA                                LSA
 25    0013          DIS3 =   $F9                                DATA
 26    0013          PADD =   $1741                PIA DATA DIRECTION REG

 28    0013          ;         ****** LOOK UP TABLES ******

 30    0013          * = $20

 32    0020          ;         TABLE FOR DAYS IN MONTHS

 34    0020   00     MOTBLE    .BYTE               0,$31,$28,$31,$30,$31,$30,$31,$31
 34    0021   31
 34    0022   28
 34    0023   31
 34    0024   30
 34    0025   31
 34    0026   30
 34    0027   31
 34    0028   31
 35    0029   30               .BYTE               $30,$31,$30,$31
 35    002A   31
 35    002B   30
 35    002C   31
 36    002D   31               .BYTE               $31,$29,$31,$30,$31,$30,$31,$31
 36    002E   29
 36    002F   31
 36    0030   30
 36    0031   31
 36    0032   30
 36    0033   31
 36    0034   31
 37    0035   30               .BYTE               $30,$31,$30,$31
 37    0036   31
 37    0037   30
 37    0038   31

 39    0039          ;TABLES FOR CONVERTING DAYS TO PERCENTAGE
 40    0039          ;NUMBERS RANGE FROM 0 TO 99
 41    0039          ;0 = LOWEST POINT 99 = HIGHEST POINT

 43    0039   50     PHY       .BYTE               $50,$63,$76,$87,$94,$98,$99,$97
 43    003A   63
 43    003B   76
 43    003C   87
 43    003D   94
 43    003E   98
```

```
43   003F  99
43   0040  97
44   0041  91              .BYTE     $91,$82,$70,$57,$43,$30,$18,$9
44   0042  82
44   0043  70
44   0044  57
44   0045  43
44   0046  30
44   0047  18
44   0048  09
45   0049  03              .BYTE     3,0,1,6,$13,$24,$37
45   004A  00
45   004B  01
45   004C  06
45   004D  13
45   004E  24
45   004F  37

47   0050  50       EMO    .BYTE     $50,$61,$72,$81,$89,$95,$98,$99
47   0051  61
47   0052  72
47   0053  81
47   0054  89
47   0055  95
47   0056  98
47   0057  99
48   0058  98              .BYTE     $98,$95,$89,$81,$72,$61,$50,$39
48   0059  95
48   005A  89
48   005B  81
48   004C  72
48   005D  61
48   005E  50
48   005F  39
49   0060  28              .BYTE     $28,$19,$11,5,1,0,1,5
49   0061  19
49   0062  11
49   0063  05
49   0064  01
49   0065  00
49   0066  01
49   0067  05
50   0068  11              .BYTE     $11,$19,$28,$39
50   0069  19
50   006A  28
50   006B  39

52   006C  50       INT    .BYTE     $50,$59,$69,$77,$85,$91,$95,$98
52   006D  59
52   006E  69
52   006F  77
52   0070  85
52   0071  91
52   0072  95
52   0073  98
53   0074  99              .BYTE     $99,$98,$97,$93,$88,$81,$73,$64
53   0075  98
53   0076  97
53   0077  93
53   0078  88
53   0079  81
53   007A  73
53   007B  64
54   007C  55              .BYTE     $55,$45,$36,$27,$19,$12,7,3
54   007D  45
54   007E  36
54   007F  27
54   0080  19
54   0081  12
54   0082  07
54   0083  03
55   0084  01              .BYTE     1,0,1,5,9,$15,$23,$31,$41
55   0085  00
55   0086  01
55   0087  05
55   0088  09
55   0089  15
55   008A  23
55   008B  31
55   008C  41

57   008D                  * = $200

59   0200           ;        ***** SUBROUTINES *****

61   0200           ;SAVES TOTAL DAYS ALIVE IN TEMP LOCS.

63   0200  A5  08   SVTDAY LDA UTDAY
64   0202  85  10          STA UTDAYT
65   0204  A5  09          LDA HTDAY
66   0206  85  11          STA HTDAYT
67   0208  A5  0A          LDA TTDAY
68   020A  85  12          STA TTDAYT
```

```
020C  60                  RTS

020D              ;SUBROUTINE TO DO PERIOD CALCULATION
020D              ;CHECKS PERIOD WITH CONTENTS OF A REG

020D  F8          PERCAL  SED
020E  85  0E              STA TEMP1       SAVE A FOR REPEATED SUBTRACTION
0210  18          PER2    CLC             FIRST CHECK TO SEE IF DIVISION
0211  A5  11              LDA HTDAYT      CAN BE PERFORMED ADD MSBS TOGETHER
0213  65  12              ADC TTDAYT
0215  D0  0A              BNE PER1        CLOSE TO CONVERGENCE IF EQUAL
0217  A5  10              LDA UTDAYT      GET LSDS OF DAYS
0219  C5  0E              CMP TEMP1       COMPARE IT TO PERIOD
021B  10  04              BPL PER1        IF PLUS, CAN STILL DIVIDE
021D  A5  10              LDA UTDAYT      IF NOT, DONE.
021F  D8                  CLD             SO RETURN
0220  60                  RTS             WITH RESULT IN A REG

0221  38          PER1    SEC
0222  A5  10              LDA UTDAYT      GET LSDS OF DAYS
0224  E5  0E              SBC TEMP1       DO THE SUBTRACTION
0226  85  10              STA UTDAYT      STORE NEW NUMBER BACK
0228  B0  E6              BCS PER2        IF CARRY SET,NO BORROW, THEREFORE
022A  38                  SEC             NO CORRECTION NEEDED
022B  A5  11              LDA HTDAYT      ELSE, DEC HUNDREDS & THOUSANDS
022D  E9  01              SBC #1          BY ONE
022F  85  11              STA HTDAYT      STORE NEW RESULT
0231  B0  DD              BCS PER2
0233  38                  SEC             DEC TEN & HUNDRED THOUSANDS
0234  A5  12              LDA TTDAYT
0236  E9  01              SBC #1
0238  85  12              STA TTDAYT
023A  4C  10 02           JMP PER2

023D              ;ADD TO DAYS, CONTENTS OF A

023D  F8          ADDAYS  SED             SET DECIMAL
023E  18                  CLC
023F  65  08              ADC UTDAY       DO 1 & 10
0241  85  08              STA UTDAY
0243  90  0E              BCC ADD1        DONE IF NO CARRY
0245  A9  00              LDA #0          ELSE DO 100 & 1000
0247  65  09              ADC HTDAY
0249  85  09              STA HTDAY
024B  90  06              BCC ADD1        DONE IF NO CARRY
024D  A9  00              LDA #0
024F  65  0A              ADC TTDAY       ELSE DO 10,000 & 100,000
0251  85  0A              STA TTDAY
0253  D8          ADD1    CLD             CLEAR DECIMAL
0254  60                  RTS

0255              ;ADD 365 TO DAYS

0255  A9  99      INCYR   LDA #$99        ADD IN STEPS
0257  20  3D 02           JSR ADDAYS
025A  A9  99              LDA #$99
025C  20  3D 02           JSR ADDAYS
025F  A9  99              LDA #$99
0261  20  3D 02           JSR ADDAYS
0264  A9  68              LDA #$68
0266  4C  3D 02           JMP ADDAYS

0269              ;CORRECTS VALUE IN A FROM DECIMAL TO HEX

0269  F0  0B      CORDEC  BEQ COR1        IF ZERO LEAVE
026B  A0  00              LDY #0          ELSE DO CORRECTION
026D  F8                  SED
026E  38                  SEC
026F  C8          COR2    INY             INCREASE Y IN HEX BY ONE
0270  E9  01              SBC #1          FOR EACH DECREASE OF A IN DECIMAL BY ONE
0272  D0  FB              BNE COR2
0274  D8                  CLD
0275  98                  TYA             RETURN WITH RESULT IN A
0276  60          COR1    RTS

0277              ;DISPLAY ROUTINE

0277  A9  7F      DISP    LDA #$7F        SET UP PIA FOR DISPLAY
0279  8D  41 17           STA PADD
027C  A2  09      DIS5    LDX #9          DIGIT INDEX
027E  A0  03              LDY #3          BYTE INDEX
0280  B9  F8 00   DIS4    LDA DIS3-1,Y    FETCH DATA
0283  4A                  LSR A           SHIFT MSNYBBLE TO LOWER POSITION
0284  4A                  LSR A
0285  4A                  LSR A
0286  4A                  LSR A
0287  20  48 1F           JSR CONVD       CONVERT AND DISPLAY
028A  B9  F8 00           LDA DIS3-1,Y    FETCH DATA FOR LSNYBBLE
028D  29  0F              AND #$F         MASK OFF MSN
028F  20  48 1F           JSR CONVD       CONVERT AND DISPLAY
0292  88                  DEY             DECREMENT BYTE INDEX
0293  D0  EB              BNE DIS4
0295  F0  E5              BEQ DIS5        STAY IN THIS LOOP UNTIL RESET

0297              ;CHECKS YEARS FOR LEAP YEAR, IF FOUND SETS FLAG.
0297              ;FLAG CAUSES USE OF SECOND HALF OF MOTBLE
```

```
0297  F0 03     LPYR    BEQ LPYR1      IF 00 SKIP FIRST CHECK
0299  4C A0 02          JMP CHECK
029C  A5 0F     LPYR1   LDA TEMP2      NOW DO MOST SIG YEARS
029E  F0 16             BEQ LPYR3      CHECK FOR YEAR '0', NOT A LY
02A0  29 1F     CHECK   AND #$1F       MASK OFF DON'T CARES
02A2  F0 10             BEQ LPYR2      CHECK FOR 20,40,60,80— ALL ARE LY
02A4  C9 04             CMP #4         FIRST TWO COMPARES CHECK IF LEFT
02A6  F0 0C             BEQ LPYR2      DIGIT IS A MULTIPLE OF TWO AND
02A8  C9 08             CMP #8
02AA  F0 08             BEQ LPYR2
02AC  C9 12             CMP #$12       NEXT TWO CHECK IF LEFT DIGIT IS ODD
02AE  F0 04             BEQ LPYR2      AND RIGHT DIGIT IS A 2 OR 6
02B0  C9 16             CMP #$16
02B2  D0 02             BNE LPYR3
02B4  E6 0D     LPYR2   INC LYFLG      IF HERE, IT IS A LY, SET FLAG
02B6  60        LPYR3   RTS

02B7                    ;CHECKS IF LYFLG SET. IF SET ADDS 12 TO CAUSE
02B7                    ;USE OF SECOND HALF OF MOTBLE

02B7  48        LYCORR  PHA            HOLD A
02B8  A5 0D             LDA LYFLG
02BA  F0 05             BEQ LYC1       IF ZERO SKIP ADD
02BC  18                CLC
02BD  68                PLA
02BE  69 0C             ADC #12
02C0  48                PHA            DUMMY PUSH FOR NEXT PULL
02C1  68        LYC1    PLA
02C2  60                RTS

02C3                    ;        ****** MAIN PROGRAM ******

02C3  A2 FF     START   LDX #$FF       SET UP STACK
02C5  9A                TXS
02C6  A9 00             LDA #0         CLEAR OUT DAY LOCATIONS
02C8  85 08             STA UTDAY
02CA  85 09             STA HTDAY
02CC  85 0A             STA TTDAY
02CE  85 0D             STA LYFLG      ZERO FLAG

02D0                    ;DAYS ALIVE CALCULATION, DO YEARS FIRST

02D0  F8                SED            SET DECIMAL
02D1  38                SEC            SET CARRY FOR SUBTRACTION
02D2  A5 07             LDA PLYR       GET PROJECTED YEAR LSDS
02D4  E5 03             SBC BLYR       SUB BIRTH YEAR LSDS
02D6  85 0C             STA TMPLYR     STORE IT AT TEMP YEARS
02D8  A5 06             LDA PMYR       NEXT DO MSDS
02DA  E5 02             SBC BMYR
02DC  85 0B             STA TMPMYR
02DE  05 0C             ORA TMPLYR     SET UP A YEAR FLAG
02E0  85 0E             STA TEMP1      IF NOT ZERO,WILL DO NORMAL CALC
02E2  D8                CLD            CLEAR DECIMAL
02E3  F0 3B             BEQ DAC1       IF ZERO SKIP YEARS CALCULATION

02E5                    ;CONVERT TOTAL YEARS TO DAYS

02E5  A5 0B             LDA TMPMYR     CHECK IF HUNDREDS SET
02E7  F0 19             BEQ DAC4       GOTO TENS AND UNITS IF NOT
02E9  A5 0C             LDA TMPLYR     CHECK IF EXACTLY 100 YEARS
02EB  D0 04             BNE DAC3       IF NOT ZERO, DO 100
02ED  A2 63             LDX #99        IF ZERO, DO 99 YEARS
02EF  D0 02             BNE DAC5       BRANCH ALWAYS
02F1  A2 64     DAC3    LDX #100       SET X FOR REPEATED LOOP
02F3  20 55 02  DAC5    JSR INCYR      ADD 365 TO DAYS
02F6  CA                DEX
02F7  D0 FA             BNE DAC5
02F9  A9 25             LDA #$25       CORRECT FOR 25 LEAP YEARS
02FB  20 3D 02          JSR ADDAYS     FOR ONE CENTURY
02FE  C6 0B             DEC TMPMYR     DEC 100 & 1000 YEARS
0300  D0 EF             BNE DAC3       MORE TO GO, IF NOT ZERO
0302  A5 0C     DAC4    LDA TMPLYR     DO 10 & 1 NOW
0304  F0 1A             BEQ DAC1
0306  20 69 02          JSR CORDEC     CORRECT FROM DECIMAL TO HEX
0309  85 0C             STA TMPLYR     REPLACE VALUE IN TEMP
030B  A2 03             LDX #3         LOOP INDEX FOR LEAP YEARS, FOR FIRST PASS
030D  C6 0C     DAC0    DEC TMPLYR     CORRECT TMPLYR FOR 1ST & LAST YEARS LIVED, ON FIRST PASS
030F  F0 0F             BEQ DAC1       DONE WHEN 0
0311  20 55 02          JSR INCYR      INC DAYS BY 365
0314  CA                DEX
0315  D0 F6             BNE DAC0       IF 0 THEN CORRECT FOR LEAP YEAR
0317  A9 01             LDA #1
0319  20 3D 02          JSR ADDAYS
031C  A2 04             LDX #4         LOOP INDEX FOR REPEATED PASSES
031E  D0 ED             BNE DAC0       BRANCH ALWAYS

0320                    ;NEXT DO DAYS FOR THE FIRST MONTH

0320  A5 02     DAC1    LDA BMYR
0322  85 0F             STA TEMP2      LOCATE FOR USE IN LY CHECK
0324  A5 03             LDA BLYR
0326  20 97 02          JSR LPYR       CHECK IF A LEAP YEAR
0329  A5 00             LDA BMON       GET BIRTH MONTH
032B  20 69 02          JSR CORDEC     CONVERT IT TO HEX
032E  20 B7 02          JSR LYCORR     CORRECT IF NEEDED
```

```
0331   AA              TAX               X= CORRECT VALUE IN HEX OF BIRTH MONTH
0332   A5  0E          LDA TEMP1         YEAR FLAG, IF NOT ZERO, DO
0334   D0  12          BNE DAC10         NORMAL CALC.
0336   A5  04          LDA PMON          ELSE, CHECK MONTHS
0338   C5  00          CMP BMON          IF DIFFERENT, DO NORMAL
033A   D0  0C          BNE DAC10
033C   38              SEC               ELSE DO THIS PART
033D   F8              SED
033E   A5  05          LDA PDAY
0340   E5  01          SBC BDAY
0342   20  3D 02       JSR ADDAYS        ADD IT TO TEMP DAYS
0345   4C  A3 03       JMP BRPER         AND DO PERIOD CALC.
034B   B5  20   DAC10  LDA MOTBLE,X      GET DAYS IN BIRTH MONTH
034C   F8              SED
034E   38              SEC
034C   E5  01          SBC BDAY          A—BDAY—C* > A
034E   20  3D 02       JSR ADDAYS        ADD IT TO TOTAL DAYS

0351                ;      NEXT DO REST OF MONTHS IN FIRST YEAR

0351   A5  0E          LDA TEMP1         YEAR FLAG
0353   F0  04          BEQ DAC15         IF ZERO, DO SHORT YEAR
0355   A9  0D          LDA #13           ELSE SET UP FOR REST OF YEAR
0357   D0  05          BNE DAC18         NORMAL BRANCH ALWAYS
0359   A5  04   DAC15  LDA PMON          GET PROJECTED MONTH FOR SHORT YEAR CALC.
035B   20  69 02       JSR CORDEC        CORRECT DECIMAL NUMBER TO HEX FOR LOOP
035E   20  B7 02 DAC18 JSR LYCORR
0361   85  0F          STA TEMP2
0363   E8       DAC11  INX               MOVE INDEX TO NEXT MONTH
0364   E4  0F          CPX TEMP2         AT END YET
0366   F0  08          BEQ DAC12         IF EQUAL, YES
0368   B5  20          LDA MOTBLE,X      ELSE, GET DAYS FOR XTH MONTH
036A   20  3D 02       JSR ADDAYS        ADD IT TO TOTAL DAYS
036D   4C  63 03       JMP DAC11         CONTINUE, COMPARE CAUSES EXIT FROM LOOP

0370                ;      NEXT DO PROJECTED YEAR'S DAYS

0370   A5  0E   DAC12  LDA TEMP1         YEAR FLAG
0372   F0  2A          BEQ DAC13         IF ZERO, SKIP THIS PART
0374   A9  00          LDA #0
0376   85  0D          STA LYFLG         CLEAR FLAG
0378   A5  06          LDA PMYR
037A   85  0F          STA TEMP2
037C   A5  07          LDA PLYR
037E   20  97 02       JSR LPYR          CHECK FOR LY
0381   A5  04          LDA PMON          GET PROJECTED MONTH
0383   20  69 02       JSR CORDEC        CORRECT IT TO HEX
0386   20  B7 02       JSR LYCORR
0389   85  0E          STA TEMP1
038B   A9  00          LDA #0            DO MONTHS FIRST
038D   20  B7 02       JSR LYCORR
0390   AA              TAX
0391   E8       DAC14  INX
0392   E4  0E          CPX TEMP1         DONE WITH MONTHS WHEN
0394   F0  08          BEQ DAC13         EQUAL
0396   B5  20          LDA MOTBLE,X      ELSE, GET NUMBER OF DAYS/MONTH
0398   20  3D 02       JSR ADDAYS        ADD THEM TO TOTAL
039B   4C  91 03       JMP DAC14

039E                ;      NEXT THE DAYS IN THE LAST MONTH

039E   A5  05   DAC13  LDA PDAY
03A0   20  3D 02       JSR ADDAYS

03A3                ;UT,HT,TT DAYS NOW CONTAINS THE AMOUNT OF DAYS
03A3                ;LIVED FROM BIRTH TO PROJECTED IN DECIMAL
03A3                ;IT IS NOW TIME TO CALCULATE THE BIORHYTHM PERIODS

03A3   20  00 02 BRPER JSR SVTDAY        FIRST SAVE TOTAL DAYS
03A6   A9  23          LDA #$23          DO PHYSICAL FIRST
03A8   20  0D 02       JSR PERCAL        PERIOD CALCULATION SUBROUTINE
03AB   20  69 02       JSR CORDEC
03AE   AA              TAX
03AF   B5  39          LDA PHY,X         GET NUMBER FROM TABLE
03B1   85  FB          STA DIS1          STORE IN DISPLAY LOCATIONS
03B3   20  00 02       JSR SVTDAY        NEXT EMOTIONAL
03B6   A9  28          LDA #$28
03B8   20  0D 02       JSR PERCAL
03BB   20  69 02       JSR CORDEC
03BE   AA              TAX
03BF   B5  50          LDA EMO,X
03C1   85  FA          STA DIS2
03C3   20  00 02       JSR SVTDAY
03C6   A9  33          LDA #$33          AND INTELLECTUAL
03C8   20  0D 02       JSR PERCAL
03CB   20  69 02       JSR CORDEC
03CE   AA              TAX
03CF   B5  6C          LDA INT,X
03D1   85  F9          STA DIS3
03D3                   JMP DISP          DONE GOTO DISPLAY ROUTINE

03D6                .END
NUMBER OF ERRORS =   0,  NUMBER OF WARNINGS  =    0
```

LATE-BREAKING MICROCOMPUTER PRODUCT NEWS, SOFTWARE RELEASES, MICRO CLASSIFIED ADS, NEW MICRO PERIPHERALS . . .

We cover the micro world as only a newspaper can— every week.

Monthly Magazines are great for in-depth articles and detailed technical information – and we're not suggesting you stop reading this one. What we *are* suggesting is that we can give you a lot that a magazine can't.

First, we're a newspaper – which means we cover a lot of things briefly, instead of a few things deeply.

Second, we're weekly – so you won't be saying "I wish I had known about that" when you hear someone talking about new micro hardware and software.

Third, we have a low-cost Microcomputing Classified Exchange, where you can find (or advertise) bargains on equipment or software – or look for people or information you need.

Fourth, we're into all aspects of computers; so you don't have to limit your information to micros. We cover everything from industry news to data processing to ethical issues, as only a newspaper can.


COMPUTERWORLD
THE NEWSWEEKLY FOR THE COMPUTER COMMUNITY
New IBM Processors Flush Out Itel AS/6, But Amdahl Stands Pat

Fifth, we'll send you 52 issues for about 35¢ a week with a one-year subscription (less than half the newsstand price). Just use the form below to start your subscription coming. It's a fast way to stay ahead.

Please send me **COMPUTERWORLD** *for 1 year* RATES: U.S. $18 Canada and PUAS - $28 Europe & Middle East - $50 All Other Foreign - $65

☐ Check Enclosed
☐ Am Ex ☐ BA/Visa ☐ MC (MC Only List four digits above your name)

If charge we must have cardholder's signature ____________ Expiration Date: ________

First Initial		Middle Initial		Surname	
Your Title					
Company Name					
Send to Address					
City		State		Zip Code	

C60

Address shown is ☐ *Business* ☐ *Home*

☐ Check here if you do not wish to receive promotional mail from *Computerworld.*

COMPUTERWORLD
THE NEWSWEEKLY FOR THE COMPUTER COMMUNITY

CIRCULATION DEPT. H, 797 Washington Street, Newton, Mass. 02160

PLEASE CIRCLE 1 NUMBER IN EACH CATEGORY

BUSINESS/INDUSTRY
10 Manufacturer of Computer or DP Hardware/Peripherals
20 Manufacturer (other)
30 DP Service Bureau/Software/Planning/Consulting
40 Public Utility/Communication Systems/Transportation
50 Wholesale/Retail Trade
60 Finance/Insurance/Real Estate
70 Mining/Construction/Petroleum/ Refining
75 Business Service (except DP)
80 Education/Medicine/Law
85 Government - Federal/State/Local
90 Printing/Publishing/ Other Communication Service
95 Other ________

TITLE/OCCUPATION/FUNCTION
11 President/Owner/Partner/General Manager
12 VP/Assistant VP
13 Treasurer/Controller/Finance Officer
21 Director/Manager of Operation/Planning/ Administrative Service
22 Director/Manager/Supervisor DP
23 Systems Manager/Systems Analyst
31 Manager/Supervisor Programming
32 Programmer/Methods Analyst
41 Application Engineer
42 Other Engineering
51 Mfg Sales Representative
52 Other Sales/Marketing
60 Consultant
70 Lawyer/Accountant
80 Librarian/Educator/Student
90 Other ________

Heinz Gf. Matuschka
583 Tarragon Dr.
San Rafael CA 94903

In Search of Memory

Vandenberg Data Products 16K board reviewed

As more sources for home computer peripherals appear, competition is bringing down prices at an astonishing pace. Even better is that some of the new suppliers are more conservative in their promises and, in fact, deliver what they advertise. This, as I have found out the hard way, can save considerable frustration, which it is hard to assign a dollar amount.

Only last March, I finally broke down and bought my Processor Technology SOL-20 kit to embark on the adventure of personal computing. Since none of the advertised software was ready by the time I had completed the construction project (contrary to the promises made by PTC as well as the store that sold the equipment), I looked for alternate sources of hardware and software.

I picked up three 4K RAM boards from S. D. Sales, which were promised and arrived four weeks after I had placed and paid for the order. After I made some minor changes, necessitated by the SOL-20 start-up procedure and some problems I encountered due to my inexperience and some bent-upon-insertion IC pins, these boards worked fine.

Since I found some non-PTC software to run on my SOL, I had to get more memory. The SOL only has five slots for S-100-bus-card use, so it became desirable to obtain more memory per card than I had bought thus far.

Checking with Advanced Microcomputer Products about the 8K RAM board they advertise for $198, I was informed that they could ship it within a week, but they wanted $219. I swallowed this bitter pill, agreed to pay the additional money and received the kit ten days later.

No alterations were necessary on this board to accommodate the SOL, and other than bent IC pins, there were no problems.

In my continuing search for more versatile software for my SOL, I came across software by H&F, San Francisco, real life savers who reasonably supplied me with an extended BASIC, patched for the SOL-20 (other than a limited 5K BASIC, PTC was still busy promising, but not delivering).

Since I had meanwhile obtained a MECA Alpha I dual-drive tape mass-storage system, which required one of my five slots, I was out of available motherboard slots, yet too low on memory to run meaningful (i.e., long) programs after loading the extended BASIC. That didn't even consider the memory required if I wanted to run my mass-storage system simultaneously; nor did it consider that I might want to add other peripherals later.

The obvious answer was to go to denser memory—at least 16K per board. Unfortunately, money was also a definite consideration.

Scanning my newest issues of computer magazines I noticed an ad that looked too good to be true: 16K static RAM for the S-100, 100 percent tested ICs, each 4K addressable to any 4K slot, fully buffered bus, soldermasked and fully socketed, all in a kit for $365. That compared favorably with the $90 per 4K I had previously paid for my S. D. 4K boards!

Compared with any of the other memories of 16K and up per board, it proved to be more than $100 less per 16K; and in some instances the other memories weren't even static.

Now, personally, I don't think there is anything wrong with dynamic memory, but it is generally cheaper to buy and harder to sell. Most hobbyists have an aversion to it, even though the aversion is usually undefined.

Full of skepticism, I called Vandenberg Data Products in Santa Maria CA, the company whose ad promised that incredible combination of features and low price.

I explained that I would like to place an order, but only if I could expect an early delivery date. I emphasized that *by early I didn't mean 1978*—maybe (I really expected bad news). Vandenberg informed me that they were delivering right now, and my order would be going out with the next shipment on Monday (this was on a Thursday). The price was as advertised, and no last-minute increase was requested, as had happened on another occasion. I almost went into shock!

Oh, well, I thought, they make a great promise, as do lots of other companies, now let's see how they deliver.

Another shock. UPS delivered the kit by Wednesday noon. Anxious, yet reluctant, I opened the package, figuring that something would have to be wrong. All the parts came well packaged, the ICs in metal tubes. Enclosed was the most thorough documentation I have yet seen with any memory board, as well as the first memory test that actually worked as claimed.

Now for the worst thing I can say about the kit. The assembly manual was obviously a first draft and contained a lot of errata. In fact three of the 4.7 uF tantalum capacitors should have been installed with the polarity reversed from the way they were shown silk-screened on the board, according to one errata. According to another, however, they were supposed to be installed on the solder side, one each between power and ground on the three 4K banks furthest from the power supply. Concentrating on the first change, I didn't pay much attention to the second until two caps were already soldered in place.

The two voltage regulators for the +5 V and +12 V power supplies came in identical TO-3 type cases (similar to a car-radio output transistor), with the same type number on each. It took some careful com-

parison between these two and the various other numbers printed on them to ascertain which was the 12 V and which was the 5 V type. (That's something you definitely wouldn't want to mistake, since the 12 V unit might not survive the demand of the 5 V circuitry.) The −5 V power requirement is taken care of by a zener and the 10 mA it draws is taken from the computers −16 V supply. Ninety mA are drawn from the +16 V supply, and approximately 650 *mA from the* +8 V. The board runs *cool!*

In addition to the relatively insignificant problems listed above, the uPD410 4K bit memory chips by NEC were rather tough to install in the sockets.

Overall construction of the kit was straightforward, with the exceptions noted above, and took me just over three hours. Some may be able to do it faster, but I like to double check everything I do; time is better spent on construction than in locating errors. This practice also usually rewards me with most items working almost immediately upon completion of the assembly.

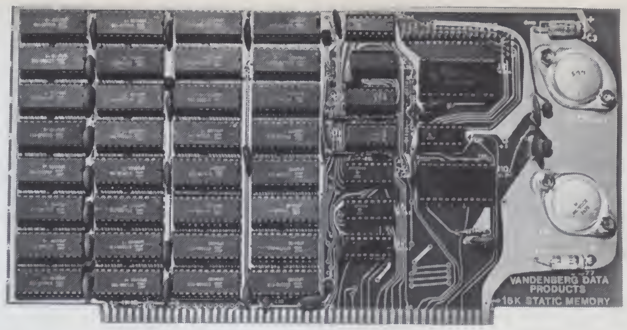
In addition, if you are careful to be immaculate with your construction, you will have fewer problems finding faulty components since you will have fewer doubts about the correct component placement and orientation.

The memory address selection is accomplished via two rows of wire-wrap pins at right angles to each other. One row of four pins, labeled from A through D, addresses the individual four 4K blocks of memory. The other row consists of 16 pins numbered in increments of four, starting with 0 and going to 60 for the starting address of the particular block. The connecting wires can either be wire-wrapped, soldered or jumpered using Molex plugs (jumpers are supplied with the kit).

Upon completion of the assembly, but prior to the insertion of the chips (as per instruction), I checked for shorts in the power supply with my ohmmeter. Finding everything in order, I plugged the board into the computer and applied power. All the voltages checked out fine.

I installed all the ICs and plugged the board into the computer again to run my first actual memory check. Perfect on the first try—not even a pin was bent.

After loading the memory check included in the manual and patching in my I/O address as indicated in the listing, I let the computer check all 32K of memory installed for about 15 minutes (I had to leave one of my 4K boards out since the Alpha I also takes up one slot).



Vandenberg Data Products' 16K static RAM board.

Since nothing worth noticing happened, I powered down, removed one of the memory chips from one of the other boards, gently bent a pin out and reinserted it. Then I started the computer and ran the memory check again.

This time, I was rewarded with a prompt from my SOL almost immediately as the checking routine, having found the error, jumped to the restart address I had patched in. I checked the error storage and found that the address as well as the data causing the failure were properly stored.

Since then, I have talked to Nelson Henderson of Vandenberg Data Products and discussed with him the major (?) problem I encountered with the errata. He informed me that Vandenberg has rewritten the manual and that all kits now include the corrected edition.

They will also include information on how the board can be wired for the phantom disable—necessary for the initialization procedure in the SOL 20, if the memory board is to be used starting at address 0.

Because of my pleasant experience in buying the 16K board from Vandenberg, I have become totally impatient with, and intolerant of, the type of wild promises that run rampant in this industry. These promises seem to be used customarily, and with a disregard for the convenience and rights of customers, even by some of the biggest companies in the business!

It's my hope that this attitude will change quickly and radically, as serious-minded companies like Vandenberg give "those others" some healthy (or unhealthy, depending on the viewpoint) competition. ■

Inventory, Accounts and Reports

Part 1: business-program design

From the forthcoming book From the Counter to the Bottom Line *by Carl Denver Warren and Thomas Richards Bailey (dilithium Press).*

The commercial availability of minicomputers since 1961, and microprocessors some 12 years later, has had profound effects on computer technology and, as a result, on data processing. The growth of personal computer technology has completely changed the computer market. No longer are computers affordable only to large companies; they have found their way into thousands of hobbyists' homes and, more recently, into the small-business environment.

The small computer is no longer a toy, but a useful tool; for a relatively modest outlay, this type of system will perform the mundane tasks of inventory control, billing and general accounting. A small-system approach to computerized accounting offers a practical application for the personal computer user. However, the growth of personal computers has been limited by a lack of application-oriented software, and little effort has been made to fill the gap. Software has, until now, consisted largely of games, number crunching, and text-editing routines.

Sucessful application program design for microcomputers must be viewed from a *top-down* approach: Emphasis is focused on the broad aspects of the application rather than the restrictions of the programming language being used.

The Automated System

Computer systems are designed to perform time-consuming tasks with speed and accuracy. Basically, a computer is task-oriented: It can do one to several jobs at a time, depending on the system, with each job made up of several smaller tasks. We are exposed to computer business applications daily, from buying breakfast at the local restaurant to purchasing airplane tickets.

What is a computerized business system? It is an automated method of handling inventory, billing, accounts receivable, accounts payable and payroll. Business systems also provide for a logical flow of bookkeeping and updating records.

Fig. 1 shows the general design of an automated system. As indicated, an automated accounting package is both a horizontal and vertical operation: Each of the major subsystems feeds the general ledger, while at the same time affecting the others. For example, the inventory-control subsystem horizontally affects the assets picture of the general ledger, and concurrently affects the accounts-payable subsystem. This in turn generates figures for the general ledger under liabilities and equity and, of course, produces the bottom line, or profit/loss picture. As you can see, an accounting package is interactive, each part dependent upon the other.

So, we can define a computerized business system as a collection of subsystems designated by task, working together to provide stand-alone and integrated totals representing business activity.

Referring again to Fig. 1, look at the makeup of each subsystem as a separate entity and how they basically interact. The first subsystem is inventory control, which is really two smaller systems—purchasing and inventory. Together they make up the inventory-control package. From the chart, we see that inventory must be purchased, thus creating expenses that can be grouped under the heading of the cost of doing business.

At the same time, an assumption is made that if you are buying inventory, you are planning to sell it. Consequently, this moves us into the accounts receivable/billing package. Accounts receivable are a direct result of sales. Sales create billings, which in turn generate receivables in your customer accounts. The inventory and accounts receivable grouped together become a major portion of current assets, which feed the asset portion of the general ledger.

Remember that when the inventory was purchased and

received, assets were acquired. Now you owe somebody something; you have accounts payable. Together, expenses, borrowing and the purchase of assets create liabilities, which are also dealt with in the general ledger. When revenue coming in versus expenses going out is calculated, the bottom line is created. This becomes part of the equity of the company, either plus or minus.

Defining the User

The businessman who needs to keep track of inventory and determine which items are losers and which are gainers, or the one who spends extra time bringing his books up to date to analyze daily transactions, can improve efficiency with a small-business accounting package.

Basically, the user of a small-business system is anyone who has decided that his bookkeeping is time-consuming enough to require automation. The following guidelines should be used when determining whether you are in need of automation: (a) inventory is large or varied; (b) receivables consist of both over-the-counter cash and billing; (c) payables are frequent enough to invite automation; (d) the general bookkeeping is difficult to handle within normal hours; (e) payroll is time-consuming and manually calculated.

If, after reviewing your situation, you find that you have one or more of these problems, and that the cost of a computer system does not overshadow the time-saving and additional effectiveness of evaluating and managing your business, then it is a good idea to automate.

The worst thing that you, the small-business owner, can do is to purchase a computer and the necessary software packages thinking this will automatically solve your problems. *It will not.* It will, however, help you pinpoint the areas that need to be watched and help in the management of these problems.

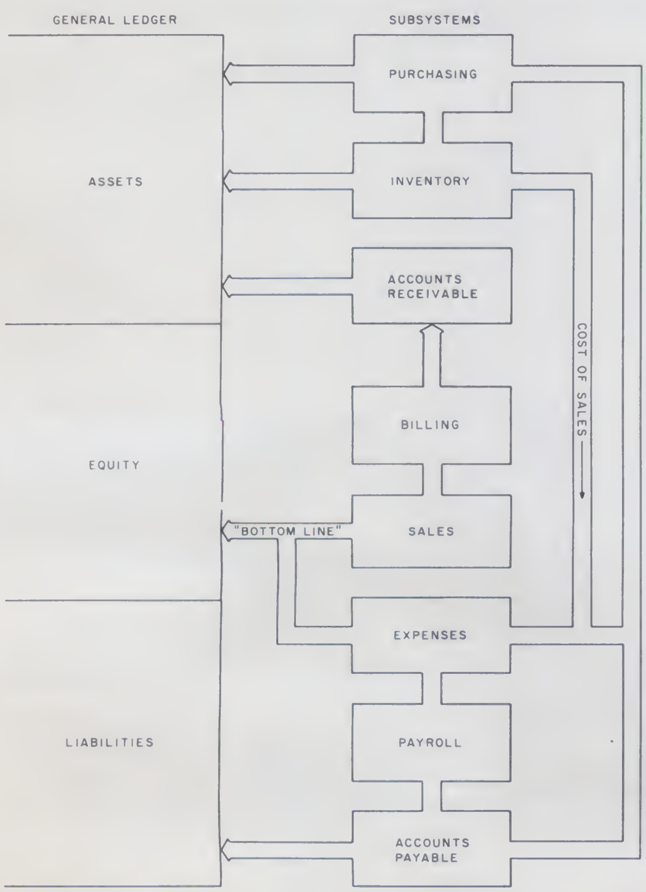


Fig. 1. General makeup and flow of an automated system.

Benefits of Automation

A microcomputer system can provide a small business with the benefits of a computerized accounting system normally affordable only to large companies. Small computers allow the use of many high-level techniques that normally only much larger companies use for handling day-to-day business.

First, by using an automated system, you will be able to process information in time to make projections and plans necessary to the well-being of your business. Next, a small system provides automated billing. Billing is usually one of the more time-consuming tasks: it requires creating the bill, then booking and handaging receivables. The automated system is designed to provide a data base for billing, calculate dates for aging receivables and reduce the work load. After automating billing and receivables, the next step is to update the receivables ledger.

Automated systems also simplify preparation of financial statements, which can be formatted in any row-column structure that fits your specific needs.

Let's review the benefits of an automated system:

- Ability to plan.
- Automated billing/accounts receivable.
- Automated accounts-receivable ledger.
- Automated payroll.
- Ease in preparation of financial statements.
- Speed and accuracy. (One word about accuracy: The computer is dumb. It will only respond to input. The old saying, "garbage in, garbage out," holds true. Error messages are provided as a guide in all computer systems, but the user must take time to learn the capabilities and needs of the system in order to achieve the greatest benefit from it.)

The automated system will also help minimize small-business failures. Many small businesses fail because they don't understand what is happening and why, and because they lack sufficient information to analyze problems and prevent them. This is not to say that the computer will analyze volumes of data and make the decisions necessary to keep you in business, nor will it create some wonderful idea that will save an already dying company.

Let's see how an automated system can help you prevent financial disaster. It will provide information about inventory, indicate reorder points, pinpoint slow and fast movers, and age the accounts receivable, showing who is late, so you can take appropriate action. The system will determine if you are operating at a profit or loss, and the causes of either condition. Equipped with this information, you can make better-informed management decisions, and develop the procedures necessary to achieve profitability or to maintain it. Remember, only you can do the decision making; the machine doesn't have that capability, but it can provide the information you need.

Design Constraints

Up to this point, our discussion of the automated accounting package has been somewhat general in order to show the complexity of the application. However, with this information in mind, we can now discuss the design constraints of an automated accounting package.

We have noted that each part of the accounting package is a subsystem of the whole, and each is interactive with the other. Each subsystem stands alone and produces totals that have meaning by themselves. Fig. 2 shows how each of the subsystems interacts with the others, and at the same time produces meaningful totals that we call reports. There are four subsystems within our

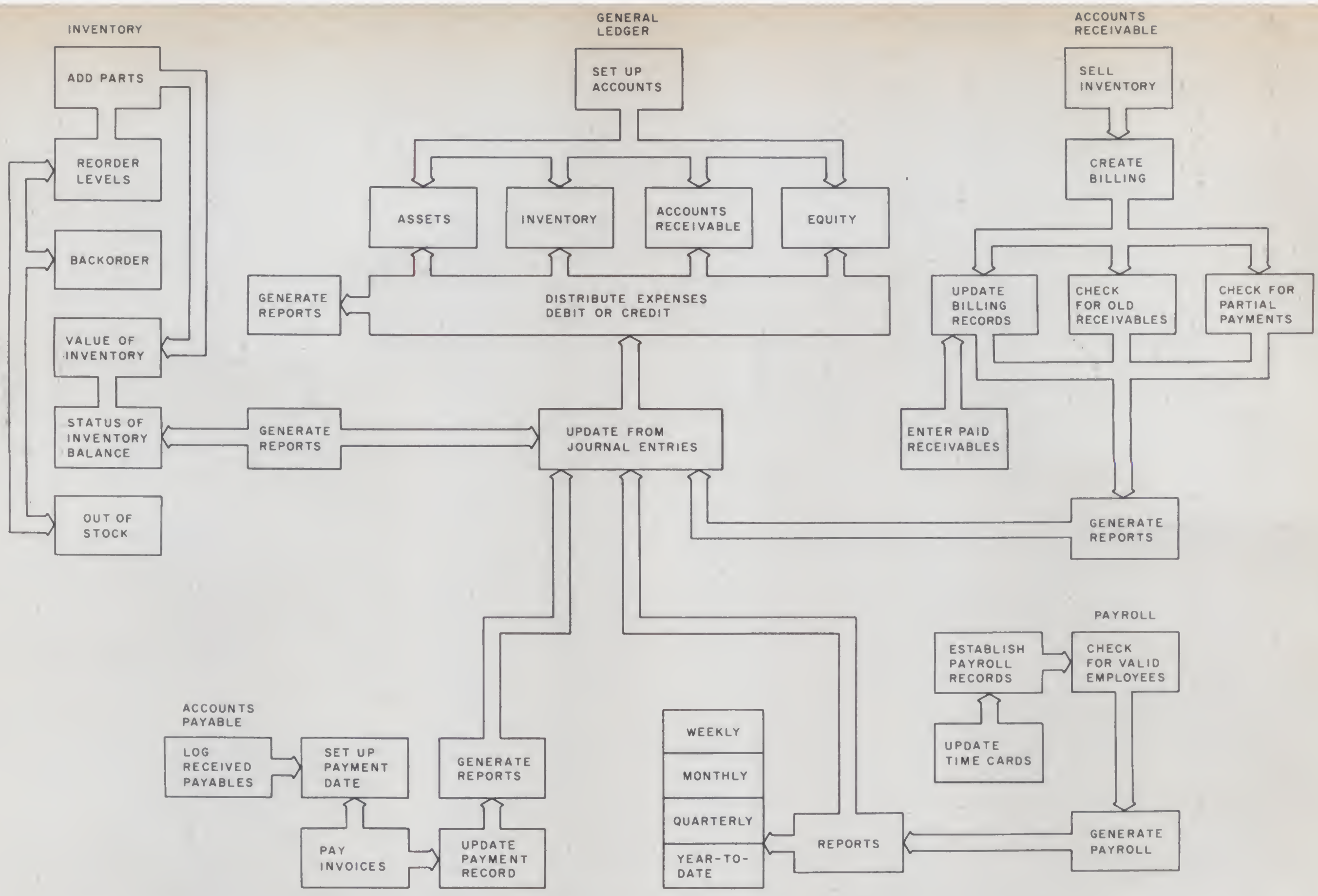


Fig. 2. Interactive block diagram.

system, and all of them come together at a central point, the general ledger.

In the design of an accounting package, each subsystem must be handled as a separate entity capable of producing usable information. Also, other decisions must be made based on the type of hardware configuration and the language used. For our purposes, each subsystem is written in BASIC. Consequently, the hardware becomes the major constraint. Large-system users with full memory expansion and dual disks have no problem. The application becomes completely integrated, with each subsystem transferring automatically to the next required process.

The *minimal* system, which we have designed our accounting system for, is a 17K machine with a video terminal, electrosensitive printer and dual digital cassette system as the mass storage device. Even though this restricts us to supplying only the bare-bones information, it is practical and unique. The only subsystem that is not feasible on this configuration is the payroll package, which would require a disk system and an addtional 32K of memory. Fig. 2 represents the general flow of an automated system and all the functions that can be performed on the minimal system, the payroll excepted.

Design Considerations

When designing an accounting package, it is important to fully understand the requirements of the business. Here are some of the types of routines that must be present to have a realistic and valuable system.

First, sort routines must be incorporated, since the accounting package provides reports used for making management decisions. These reports must be in order, for example, by date, vendor or vendor number. Each subsystem in Fig. 2 requires that some type of sort take place to generate the most meaningful reports.

Next, save and load routines must be incorporated to allow for file management of fixed and variable data. For the BASIC interpreter, this means converting string arrays to numerical arrays and back again. This is particularly important on the minimal system configuration.

Flexibility must be built into the system, particularly when financial statements are being produced. Therefore, the ability to format statements in any row-column structure using simple rules is a requirement.

The automated system is handling all the mundane tasks and producing totals. These numbers should be available via a planning package to help management determine where the business is and where it is going.

An important external consideration is establishment of a document flow to feed the system. In some cases, such as billing, it may be necessary to replace existing billing forms; for the other subsystems, existing forms can frequently be used.

Summary

In this article, we have tried to give you a basic understanding of small-business accounting and some of the design criteria used for developing an accounting system. We realize that this is an overview of the total problem, but it should indicate the complexity of the application.

In a future article, we plan to describe the inventory package, present program listings and discuss procedural use. Additionally, we will discuss tape techniques as they apply to the inventory program in particular—and to the system in general.■

Laurence A. McCaig
Microtec Computers, Inc.
112 Elm St.
Newport ME 04953

Small Business Software

Part 1: accounts receivable

The author and small-business system from Microtec Computers, Inc.

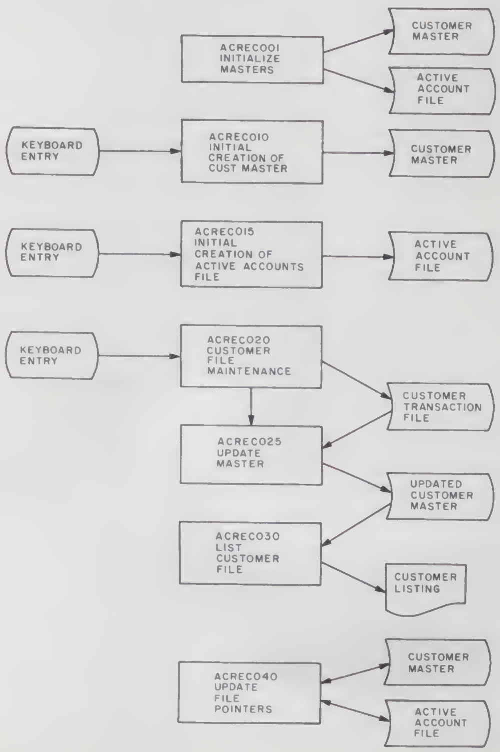


Fig. 1. Accounts Receivable program flowchart.

With the rapid advancement of microcomputer technology, more small businesses are becoming aware of the advantages of installing their own in-house computer systems. Progress has been phenomenal during the past few years in decreasing both the size and the price of these machines. There are two major reasons why many prospective users are hesitant to purchase systems. First, there are very few complete microcomputer business systems available. The businessman wants to be able to bring a system into his office, plug it in and start using it. Second, there haven't been many professional-quality applications programs available for business accounting. Many people question whether these small machines are capable of the complexities involved in many business applications. With the emergence of reliable, complete systems and software, the sales of small-business microcomputer systems will increase rapidly.

A few weeks after my first serious look at microcomputers, I ordered what I thought would be a good system at a moderate price. Then, I ordered all the peripheral components necessary for a good business system. As soon as the system was up and running, I added a BASIC interpreter, and printer I/O routine, and began writing applications programs.

Overall Objective

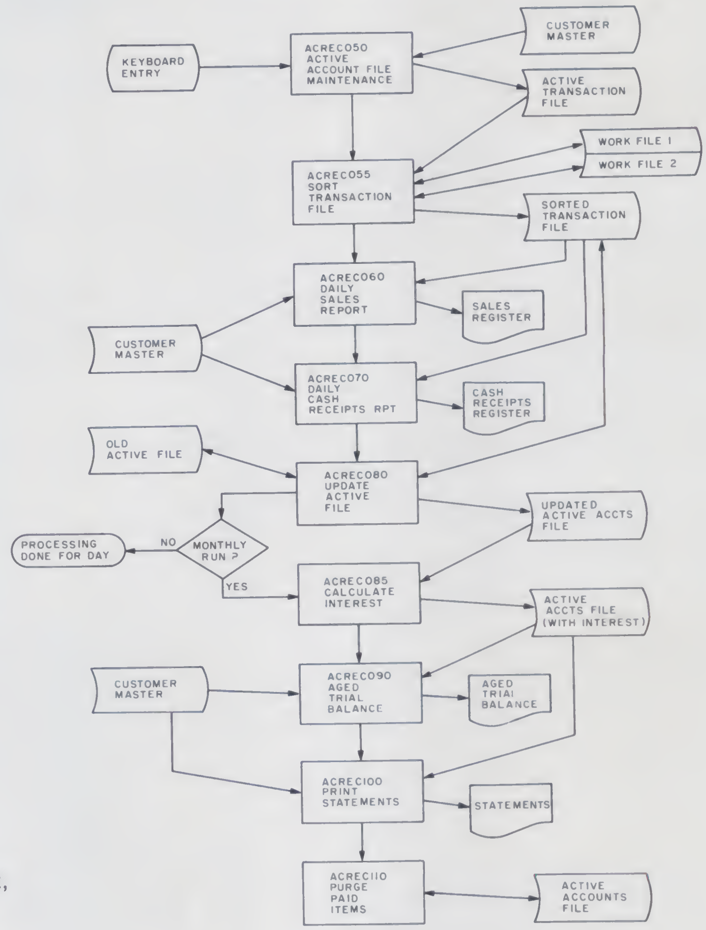
This series of articles will detail the software necessary for a business accounting package. Included will be Accounts Receivable, Accounts Payable, Inventory and Payroll programs. Output from these systems will feed directly into the general ledger system.

POS	0-4	5-9	10-19	20-36	37-58	59-80	81-90	91-95	96
CONTENTS	ACTIVE FILE POINTER	CUSTOMER NUMBER	FIRST NAME	LAST NAME	FIRST ADDRESS LINE	SECOND ADDRESS LINE	TELEPHONE NUMBER	MONTHLY PAYMENT	END MARK
PROGRAM NAME	A	F	A$	B$	C$	D$	E$	G	

(a)

Customer Master File

Customer number =4 digits numeric only.
First name =8 characters.
Last name =15 characters.
Adrs line 1 =20 characters.
Adrs line 2 =20 characters.
Telephone # =8 characters.
Monthly pmt =up to 9,999,999.99.



(b)

Active Accounts File

Customer number =4 digits numeric only.
Date =6 digits numeric only in year, month, day format.
Invoice number =5 digits numeric only.
Part number =5 digits numeric only.
Quantity =up to 9,999.99.
Description =10 characters.
Terms code =1 character.
Amount =up to 9,999,999.99.
Sales tax =up to 9,999,999.99.
Interest =up to 9,999,999.99.
Transaction cd =1 character (valid codes are P for payment, C for charge, M for credit memo and D for debit memo).

POS	0-4	5-9	10-14	15-19	20-24	25-29	30-41	42-44	45-49	50-54	55-59	60-62	63
CONTENTS	CUSTOMER NUMBER	CUSTOMER FILE POINTER	DATE	INVOICE NUMBER	PART NUMBER	QUANTITY	DESCR.	TERMS CODE	AMOUNT	SALES TAX	INTEREST	TRANS-ACTION CODE	END MARK
PROGRAM NAME	L	M	N	O	P	Q	F$	G$	R	S	I	H$	

Fig. 2. Physical file layouts.

All systems are designed to present reports in a highly organized, businesslike manner. For example, the receivable system will contain the following reports: customer master listing, daily and monthly sales report, daily and monthly cash receipts report, aged trial balance, active account listing and statements.

The Inventory System will be linked to the Accounts-Receivable system and will include automatic invoice generation and posting to the receivable file. Items will be removed from inventory as they are sold, and updated as they are received. Flags will indicate reorder points, and warnings will be printed for any inventory items that increase in price when they are reordered. Stock status, receivings and usage reports will also be produced.

Equipment Requirements

At Microtec, we have developed a system that will run the programs as presented. Or, you may choose the PolyMorphic System 16, three North Star micro-diskette drives and a Centronics 101A printer. With 24K of memory, this system can process receivables for 800 to 1000 customers, and maintain an inventory of about 2000 items. Either system will run any of the software described in this series.

Software

Each software system will be presented in three to five installments. The Accounts Receivable system is first, followed by the Inventory system.

Fig. 1 show the elements of the Accounts Receivable system. The program ACREC050 displays all account information for any customer, and allows entry of payments, charges and credit or debit memos. This is a handy tool for immediate entry of data—or, of course, you can make the entries in the batch mode at the end of the day. The program may be terminated and restarted at any time with-

out loss of data. The active file is updated on a daily basis, so displayed information is never more than one day old. At the end of each day, the transaction file is sorted, and daily sales and cash receipts reports are printed. If these report balances agree with your manual tally, the transaction file is merged with the active account file.

At the end of each month, ACREC085 is run to calculate interest on all accounts over 30 days old. An aged trial balance is then produced and statements are printed. The statements are designed to fit a standard window envelope. After the statements are run, the active file is purged of all paid accounts, and balances are updated.

Direct-access pointers have been embedded in the active file, customer master file and active transactions file to allow rapid cross-referencing of data. This is the same approach used in very large data bases.

Refer to Fig. 4 for the following discussion. Some programs are written specifically to implement a system. Look again at Fig. 1 and note that ACREC001, ACREC010 and ACREC015 accomplish this task. These programs, along with two different random access routines, will be used in several programs of this and other systems.

The first program, ACREC001, initializes the customer master and active accounts files. Unless you have a bad habit of destroying files, it is not necessary to waste valuable disk space for this program, as it will be used only once.

During initial loading of these files, all data must be entered sequentially. This will not be required during maintenance functions as we will use the random access methods described later. I recommend that a gap of at least five, and preferably ten, numbers be left between account designations, and that all customer numbers be assigned in alphabetical order. First and last name are loaded separately, making it possible to sort on the last name if desired. I will also show you how to create secondary indexes that will allow you to access a file in any desired sequence without rearranging the file. Thus, the file can be accessed two different ways at the same time.

Now, look at Figs. 2a and 2b. These show the physical layout of the two master files that will be produced by the programs presented here. North Star BASIC is written with a precision of eight, and all numeric data is stored in floating-point format requiring

```
>LIST
10 REM *--ACREC010--
20 REM *--CUSTOMER NAME AND ADDRESS LOAD PROGRAM--
30 REM *--COPYRIGHT JULY 15,1977 BY MICROTEC COMPUTERS INC.
40 RM STREET, NEWPORT, MAINE 04953
50 REM
60 DIM B$(15)\DIM C$(20)\DIM D$(20)\A6$(1,8)="ZZ"
70 J=0\OPEN #0,"CUSTMST,2"
80 GOSUB 400
90 J=G
100 GOSUB 400
110 N=0
120 !"LAST CUST # ENTERED WAS ",F," ",A$," ",B$
130 !
140 !"TO END PROGRAM ENTER 9999 AT CUST #"
150 J=J+1
160 !"ENTER FOLLOWING:"
170 GOSUB 520
180 INPUT "CUST # ",F
190 IF F=9999 THEN 490
200 IF F<=N THEN 220
210 N=F\GOTO 240
220 !"SEQUENCE ERROR-RETYPE"
230 GOTO 160
240 INPUT "1ST NAME ",A$(1,8)
250 IF A$=A6$ THEN 380
260 INPUT "LST NAME ",B$(1,15)
270 IF B$(1,2)=A6$(1,2) THEN 380
280 INPUT "ADRS LIN1 ",C$(1,20)
290 IF C$(1,2)=A6$(1,2) THEN 380
300 INPUT "ADRS LIN2 ",D$(1,20)
310 IF D$(1,2)=A6$(1,2) THEN 380
320 INPUT "TEL # ",E$(1,8)
330 IF E$(1,2)=A6$(1,2) THEN 380
340 INPUT "MO PMT AMT ",G
350 IF G=9999 THEN 380
360 GOSUB 420
370 GOTO 150
380 N=N-1
390 GOTO 170
400 READ #0%96*J,A,F,A$,B$,C$,D$,E$,G
410 RETURN
420 WRITE #0%96*J,A,F,A$,B$,C$,D$,E$,G
430 K=0\G=J
440 WRITE #0%96*J,A,F,A$,B$,C$,D$,E$,G,NOENDMARK
450 RETURN
460 J=0
470 GOSUB 420
480 GOTO 140
490 REM *--CLOSE ROUTINE--
500 CLOSE #0
510 END
520 C$(1,20)="                    "\REM 20 BLANKS
530 D$=C$\A$=C$\B$=C$\E$=C$
540 RETURN
```

Program 1.

```
>LIST
10 REM *--ACREC001--
20 REM *--USED ONLY ONCE TO INITIALIZE THE CUSTOMER--
30 REM *--MASTER AND ACTIVE ACCOUNTS FILE--
40 REM *--COPYRIGHT JUNE 15, 1977 BY MICROTEC COMPUTERS INC. --
50 REM *--112 ELM STREET, NEWPORT, MAINE 04953
60 REM
70 DIM B$(15)\DIM C$(20)\DIM D$(20)\DIM F$(10)\DIM G$(1)
80 DIM H$(1)\J=0\K=0\O=0\G=0\I=0\DIM A$(8)\DIM E$(8)
90 OPEN #0,"CUSTMST,2"\OPEN #1,"ACTIVE"
100 INPUT "ENTER CUSTMST, ACTIVE, OR BOTH ",Z$
110 IF Z$(1,7)="CUSTMST" THEN 150
120 IF Z$(1,6)="ACTIVE" THEN 170
130 IF Z$(1,4)="BOTH" THEN 150
140 !"INVALID COMMAND-RETYPE"\GOTO 100
150 WRITE #0%96*J,A,F,A$,B$,C$,D$,E$,G
160 IF Z$(1,4)<>"BOTH" THEN 150
170 WRITE #1%63*K,L,M,N,O,P,Q,F$,G$,R,S,I,H$
180 CLOSE #0\CLOSE #1
190 END
```

Program 1a.

five bytes each. All alphanumeric data put out to disk requires two bytes plus the length of the string if less than 256 bytes, and requires three bytes if equal to or greater than 256 bytes. One byte must also be reserved for an end-of-file mark.

Since we must be able to access the files randomly, it is important that all records be of fixed length. Fig. 2a shows the layout of the customer master file. Please follow the recommended sizes for customer numbers or problems will arise later on when you format reports. The same applies to customer number, date, invoice number, part number and quantity on the active file (Fig. 2b). The numbers at the top of these layouts refer to physical location of the data, and do not reflect theoretical size.

Look at Program 1 (ACREC010). Record number zero is reserved for storing pertinent information about all files in the system. Therefore, you must run ACREC001 (Program 1a) once to prevent a file error the first time you try to run Program 1.

Line numbers 70-130 read record zero and get the pointer of the last record on file, which is stored in G. The file then advances to this record and prints out what is necessary to let the operator know where he or she left off. Record zero is updated each time an addition is made to the file.

I tried updating only at the end of the program, and all went well until there was a power failure after I had loaded about 50 customers. Record zero still had the pointer of the last record entered on a previous run. It took about half an hour to patch up the file.

Line numbers 150-370 are

Program 2.

```
>LIST
10 REM *--ACREC015--
20 REM *--PROGRAM TO LOAD ACTIVE FILE FIRST TIME--
30 REM *--LAST POINTER STORES IN G OF RECORD ZERO--
40 REM *--COPYRIGHT JULY 2, 1977 BY MICROTEC COMPUTERS INC.
50 REM *--112 ELM STREET, NEWPORT, MAINE 04953--
60 REM
70 DIM F$(10)\DIM G$(1)\DIM H$(1)\A6$="ZZ"\L1=0
80 OPEN #0,"ACTIVE"
90 K=0\L2=0\GOSUB 500
100 GOSUB 530
110 !"LAST ACCT ",L," DT ",N," FOR ",F$," AMT ",%7F2,R
120 L1=L
130 GOSUB 500
140 K=K+1\!"ENTER FOLLOWING AS CALLED"
150 !"TO END PROGRAM TYPE 9999 AT CUST #"
160 INPUT "CUST # ",L\IF L=9999 THEN 550\IF L>=L1 THEN 180
170 !"NUMBERS OUT OF SEQUENCE-RETYPE"\GOTO 160
180 IF L=L1 THEN 210
190 INPUT "TERM CD ",G$(1,1)
200 IF G$=A6$(1,1) THEN 160
210 M=0\INPUT "DATE YYMMDD FORMAT ",N\IF N=9999 THEN 160
220 INPUT "TRAN CD ",H$(1,1)
230 IF H$=A6$(1,1) THEN 160
240IF H$="C" THEN290\IF H$="P" THEN 270\IF H$="M" THEN 290
250 IF H$="D" THEN 290\!"INVALID CODE RETYPE"
260 GOTO 220
270 F$(1,10)="PAYMENT"
280 O=0\P=0\Q=0
290 L1=L\INPUT "INVOICE # ",O\IF O=9999 THEN 160
300 IF H$="P" THEN 350
310 INPUT "PART # ",P\IF P=9999 THEN 160
320 INPUT "QTY ",Q\IF Q=9999 THEN 160
330 INPUT "DESCRIPTION ",F$(1,10)
```

```
340 IF F$(1,2)=A6$(1,2) THEN 160
350 INPUT "AMOUNT ",R\IF R=9999 THEN 160
360 IF H$="P" THEN 420
370 S=0\INPUT "Y FOR SLS TAX ",K$
380 IF K$<>"Y" THEN 410
390 S=R*.05
400 !"TAX IS ",%6F2,S
410 IF K$=A6$(1,1) THEN 160
420 I=0\GOSUB 450
430 GOSUB 500
440 GOTO 140
450 WRITE #0%63*K,L,M,N,O,P,Q,F$,G$,R,S,I,H$
460 K2=0
470 O=K
480 WRITE #0%63*K,L,M,N,O,P,Q,F$,G$,R,S,I,H$,NOENDMARK
490 RETURN
500 F$(1,10)="          "\REM 10 BLANKS
510 H$=" "
520 RETURN
530 READ #0%63*K,L,M,N,O,P,Q,F$,G$,R,S,I,H$
540 RETURN
550 CLOSE #0
560 END
```

```
>LIST
5000 REM *--RANDOM ACCESS ROUTINE
5010 REM *--COPYRIGHT JUNE 17,1977 BY MICROTEC COMPUTERS INC.
5020 REM STREET, NEWPORT, MAINE 04953
5030 REM
5040 Q9=0\IF Z6=1 THEN 5160
5050 DIM Z1(19,1)
5060 Z6=1\J=0\Z9=1
5070 GOSUB 5380\Z7=G
5080 Z8=G/20\Z8=INT(Z8)\IF Z8=0 THEN Z8=1
5090 FOR Z=0 TO 19
5100 J=Z9
5110 GOSUB 5380
5120 Z1(Z,0)=F\Z1(Z,1)=Z9
5130 Z9=Z9+Z8
5140 IF Z9>Z7 THEN Z9=Z7
5150 NEXT
5160 FOR Z=0 TO 19
5170 IF Z1(Z,0)>=A9 THEN 5180 ELSE 5210
5180 H9=Z1(Z,1)
5190 IF Z=0 THEN L9=1 ELSE L9=Z1(Z-1,1)
5200 EXIT 5230
5210 IF Z9=19 THEN EXIT 5360
5220 NEXT
5230 R9=H9
5240 GOTO 5290
5250 R9=(H9-L9)/2+L9
5260 R9=INT(R9)
5270 IF H9=R9 THEN 5360
5280 IF L9=R9 THEN 5360
5290 J=R9\GOSUB 5380
5300 IF A9=F THEN 5370
5310 IF A9>F THEN 5340
5320 H9=R9
5330 GOTO 5250
5340 L9=R9
5350 GOTO 5250
5360 Q9=1
5370 RETURN
5380 READ #0%96*J,A,F,A$,B$,C$,D$,E$,G\RETURN
```

Program 3.

the data entry statements for the file. Note that if you type ZZ when an alpha item is requested, or 9999 when a numeric item is requested, it will bring you back to the starting point and allow you to reenter the data for this customer. This is handy if you get to the last name and notice that the customer number has been entered incorrectly. If you try to enter customer numbers out of sequence, an error message will be displayed and you will be returned to start. After completing data entry, type 9999 at the customer number entry, and the end-of-file processing will be performed.

Program 2 (ACREC015) is used for creation of the active accounts file. Lines 90-110 position the file and let the operator know where he or she stopped. The pointer of the last record on file is stored in O of record zero. The same rules for error correction follow in this program: typing ZZ for alpha items and 9999 for numeric items will return you to the start position. This method will be used in all programs.

When the *terms* code is requested, type in the codes that you want to set aside for the terms you use. The system normally uses only one, A, for "Net 10 days, 1.5% over 30 days." Most small businesses will find this adequate. I will show you where modifications must be made for other terms as they are needed.

Four valid transaction codes are used: P for payment, C for charge, M for credit memo and D for debit memo. Anything else will produce an error message.

Next, the following information is entered; invoice number, part number, quantity, description, amount, and, if applicable, Y is entered after the amount. The sales tax will then be calculated and displayed. Line 390 may have to be changed, depending on your state's sales-tax rate.

If P is entered for the transaction code, a shortcut is taken and only invoice number and amount will have to be

entered. Note that in both programs all strings are defined as a definite length; this is necessary if random accessing is to be done. Also note that the date is entered in year-month-day format, making aging and interest calculations easier.

Program 3 is a subroutine to read the customer master file randomly. It first takes the number of records on file and divides by 20. The 20 segments of the file are then loaded into a table containing the customer number and file pointer of each record. (This is done only the first time a request for a customer record is found.) Each time a request for a particular customer is made, this table is searched until a customer number equal to or greater than the one requested is found. The pointer of this record is stored in H9 (high file pointer), and the pointer of the previous entry in the table is stored in L9 (low file pointer).

Next, a search routine is entered that divides the area on file by two until the desired record is either found or proven not on file, in which case Q9 will be set to 1 (see Fig. 3). If the file contained 200 entries, the first ten entries in the table might look like Fig. 3a.

Suppose that we are looking for customer number 1567. First, the table would be searched to item number five, where customer number 1927 would be found, located at record number 50. Now H9 will be loaded with pointer number 50, L9 with pointer 40. Fig. 3b shows what the file might look like from record number 40 to record number 50. Now, record 50 is checked; it is not the customer that has been requested, so the low pointer (40) is subtracted from the high pointer (50), and the result (10) is divided by two. The low pointer value in L9 is then added to this for a new pointer value of 45—the next record checked.

Here, we find that the customer number is less than the one desired, so L9 is replaced with 45. This process continues, reading records 46, 47, 48 and 49, where the desired record is found. Using this method, the file would be read no more than five times to get any one record. This method requires about 400 more bytes than Program 3a, but it is much faster and well worth the memory.

Program 3a operates much the same way as Program 3 except that there is no table in memory, and the entire file must be searched each time a request is made.

Go ahead and try these subroutines. Be sure to open the file and set up the proper dimension statements. Enter the customer number desired and execute a GOSUB 5000. After this, check the contents of Q9; if it is equal to 1, the record you want has not been found.

Next time, we'll cover the maintenance programs and the sort that goes with ACREC050. I will be using the random access routines in these programs. If you try them, you will understand their usefulness.

Entering your customer and account data should keep you busy until then.■

Contents Item	Customer #	File Pointer
0	398	10
1	675	20
2	759	30
3	1134	40
4	1927	50
5	2468	60
6	2875	70
7	3296	80
8	4015	90
9	4659	100

a)

Record #	Customer #
40	1298
41	1300
42	1316
43	1317
44	1321
45	1423
46	1490
47	1492
48	1500
49	1567
50	1927

b)

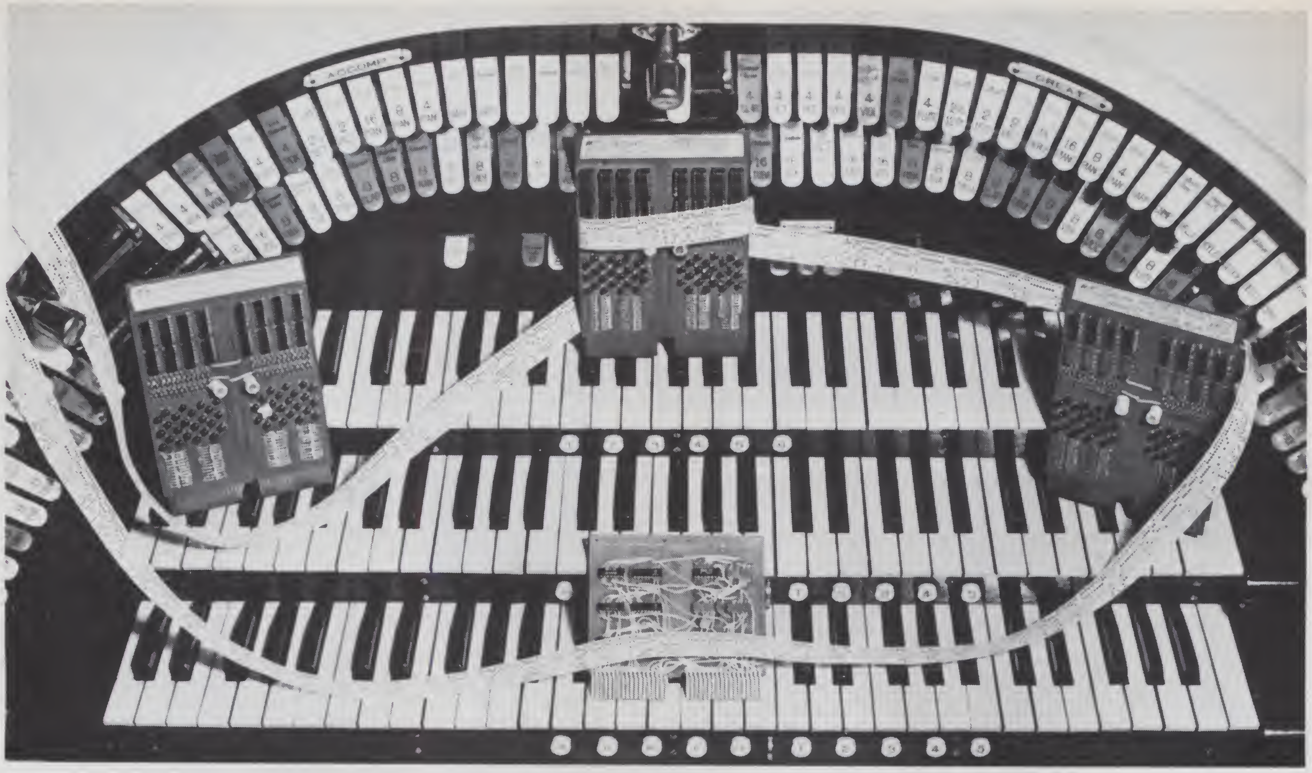
Fig. 3.

Program No.	Program Name	Purpose
1a	ACREC001	Initializes the Customer Master and Active files.
1	ACREC010	Customer file creation program.
2	ACREC015	Active Account file creation program.
3	-	Fast random file access routine.
3a	-	Alternate random file access routine.

Fig. 4. Customer file programs.

```
>LIST
5000 REM *--RANDOM ACCESS ROUTINE--
5010 REM *--COPYRIGHT JUNE 17,1977 BY MICROTEC COMPUTERS INC.
5020 REM *--112 ELM STREET, NEWPORT, MAINE 04953
5030 REM
5040 J=0\Q9=0
5050 GOSUB 5220
5060 H9=G\L9=0
5070 R9=H9
5080 GOTO 5130
5090 R9=(H9-L9)/2+L9
5100 R9=INT(R9)
5110 IF H9=R9 THEN 5200
5120 IF L9=R9 THEN 5200
5130 J=R9\GOSUB 5220
5140 IF A9=F THEN 5210
5150 IF F<A9 THEN 5210
5160 H9=R9
5170 GOTO 5090
5180 L9=R9
5190 GOTO 5090
5200 Q9=1
5210 RETURN
5220 READ #0%96*J, A, F, A$, B$, C$, D$, E$, G
5230 RETURN
```

Program 3a.



Wurlitzer console at the Crown Theatre, Pasadena CA. (Los Angeles Times Photo)

The Music Man

Prentiss Knowlton's computerized pipe organ

Sheila Clarke
CyberGrafix
518 N. Brand
Glendale CA 91203

Wandering through the Personal Computing Show in Los Angeles last May, I made my way through the din of beeping, blinking cursors, the clack of printers and click of floppy disks. Then my ears perked up at the sound of music: not the usual tinkly emanations—or sounds that closely resemble tunes plucked from the teeth of combs that are generally referred to as computerized music. I was hearing Bach's "Concerto in A Minor," played with perfection on a theater pipe organ. *Computerized?* I was drawn first to the stereo and album propped against it, then, with an incredulous look, to the tall, slim man standing behind the counter of the booth.

Prentiss Knowlton does not fit the image I've come to associate with programmers and engineers . . . more the image of an artist. He is taller than average, and his large slender hands look as if they were meant to play music. When he speaks to you, he looks directly into your eyes, and when you answer, he reacts with intense interest. You quickly discover his range of interest covers system programming, playing the organ and, ultimately, controlling his own pipe organ with a computer. I was intrigued by his goal: to control the organ to such a degree that it would produce the range of sensitivity and expression that a musician would put into his own music.

Prentiss and I spent a good deal of time in subsequent months discussing his theories, philosophy and background. His achievements ought to be shared.

For a bit of background: he holds a PhD in Computer Sciences and is a systems analyst at Jet Propulsion Laboratories in Pasadena CA. His musical avocation began when he was a student at the University of Utah. While an undergraduate, he studied organ, with an emphasis on performing rather than composing. When one of his classes required a special project, he first conceived a graphic display of traditional musical scores which would be derived through interaction by the composer, using the computer as a tool. This would relieve the composer of clerically annotating original music.

To accomplish this, he first bought an organ in kit form, which enabled him to understand the instrument's working parts. He then interfaced the organ's keyboard directly to a computer; when programmed, it resulted in a printout of originally composed music. The project culminated in his PhD dissertation, and was published at the University of Utah in August 1971. If you're interested in computer output of musical notation, read Dr. Knowlton's article, "Capture and Display of Keyboard Music" (*Datamation,* May 1972).

Exploration of computerized organ music continued when Prentiss returned home to California and renewed his project after purchasing a used pipe organ containing seven ranks. A rank is a set of wind pipes, which can range in number from 12 to 90 pipes, contained in a windchest. Several problems arose, which had to be solved before he progressed further.

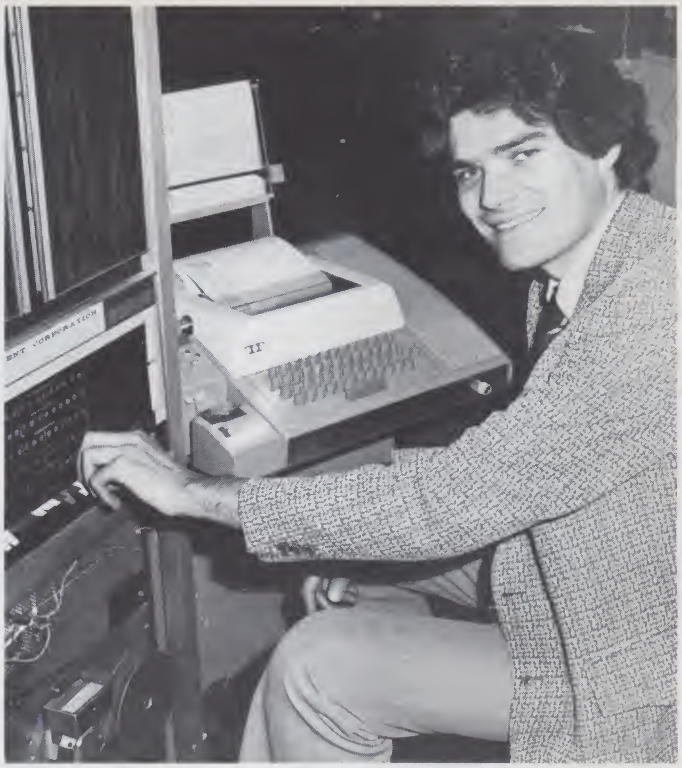
You should know that when a pipe organ is installed, usually in a church or theater, it is considered to be a permanent installation. Prentiss, however, had to contend with moving, storing and assembling his organ several times and had to find a way to cut down the time involved—normally many days. Since he wanted an instrument that could go on tour, he designed a method of labeling and crating the parts so the entire system would fit into the back of a pickup truck. Now the operation takes no more than three hours.

Another problem involved avoiding destruction of the electronic connectors during disassembly. The solution turned out to be the use of three 25-pin connectors per pipe . . . the same used on a typical data terminal (see photo). Unplugging the connectors and detaching the legs makes a simple knock-down operation. Each chest of pipes functions independently from the others and requires its own windline. Prentiss economized when he discovered that dryer hose purchased from a department store worked well. "I have about $30,000 invested so far, which includes about $8000 in the organ alone," he says. "The going rate for a pipe organ today is about $10,000 per rank. I have about 10 ranks." That doesn't sound like it would fit the computer hobbyist's budget. But hang in there.

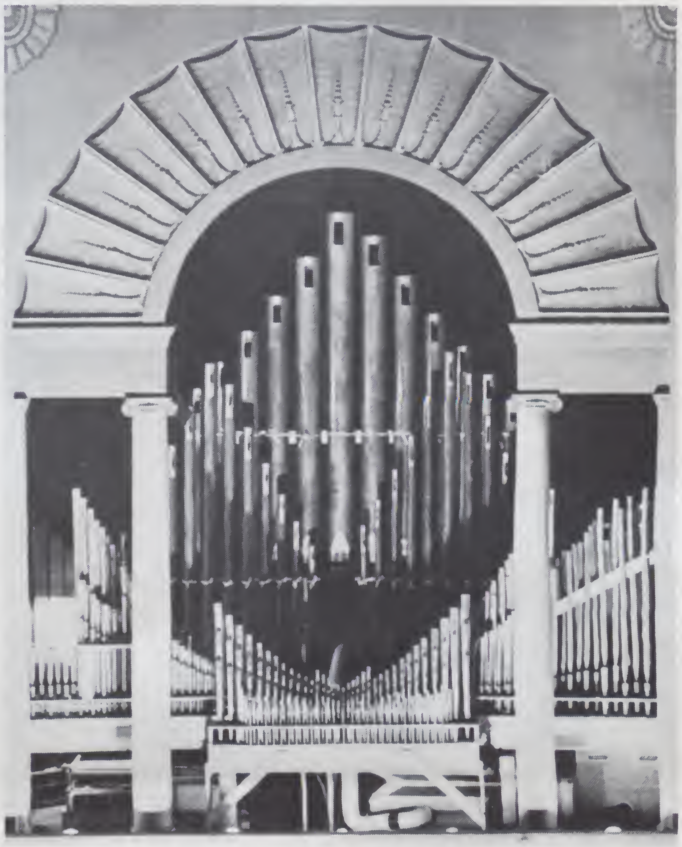
Computer Equipment

When he first interfaced the system to a computer in 1972, the organ contained seven ranks. His computer system began with a DEC PDP-8 and a Teletype, with punched paper tape for storage. Ordinary sheet music was coded using a musical description language, said to be similar to BASIC in its interpretive qualities. The language is the result of help from many collaborators, including Alan Ashton, who initially developed the music description language.

Later the computer system was upgraded to read magnetic tape, and Prentiss obtained a second PDP-8. Now, with the tape unit on the second PDP-8, another output port was made available. An

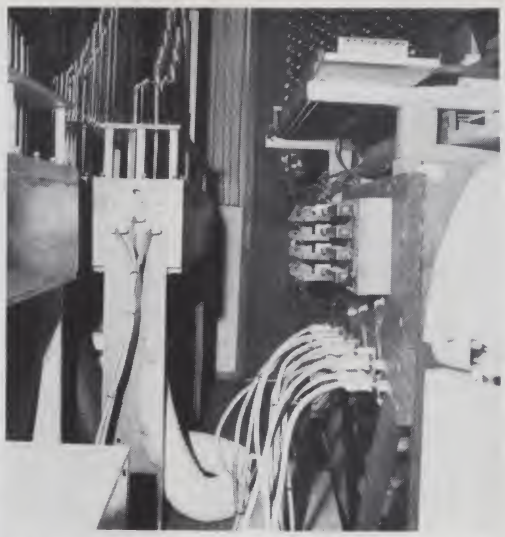


Dr. Prentiss Knowlton produces a totally automatic concert under computer control. His understated comment: "My role with the music is somewhat indirect."



Pipes for the Wurlitzer pipe organ, housed at the Crown Theatre in Pasadena CA. (Los Angeles Times Photo)

Chests for ranks of pipes were especially designed for mobility and to enable more convenient connection to the computer. The connectors are easily plugged in during reassembly of the entire system. Three connectors handle one set of pipes.



The computerized organ uses the same power supply as a traditional system. The main difference is that the wires are not soldered; removable connectors are used instead. When a key is pressed, the organ console's corresponding pipes are permitted to play through use of the connectors.

output port was added to the second PDP-8; and the two computers were connected with a serial line. The second computer ran programs on magnetic tape. Literally speaking, one computer was the organist and the other was the page turner. When the first computer needed a character, it would ask either the operator or the second computer.

I don't entirely understand yet how the two computers really interact, but getting Prentiss to explain in tutorial terms is difficult. He's been so involved for so long that, to him, it's second nature—like breathing.

During our last conversation in September, Prentiss told me that he is expanding his system to include an Imsai with 48K of memory, dual floppy disk drives from Digital Systems, a Sanyo video monitor and a VDM input keyboard. The computer will be interrupt driven by a priority interrupt controller (PIC-8) for the 8080. He'll use the Oliver Paper Tape Reader. Now he's building an interface to control a P10/4 board from the Imsai. A dual serial I/O port and S10-2 will talk to the PDP-8 for "page turning." When he adds the Diablo 1620 Hiterm Printer, he'll have invested a total of $10,000 in computer equipment. All of this will operate his own pipe organ located at the Crown Theatre in Pasadena. At last report, he intended to have the Imsai perform all computer functions.

Prentiss fully expects to develop a product for adaptation on an Altair or Imsai for hobbyist use. Since Altair bus boards that will produce electronic sound are already available, he envisions connecting one to a music keyboard and writing software for exchange with other enthusiasts. In fact, through this article, he has extended an invitation to anyone who wants to be involved. His reasoning is that the more people who are interacting, the more creative will be the results, and the more computer hobbyists and music will benefit. You can write him c/o Computer Humanities, 255 N. Madison Ave. #L, Pasadena, CA 91101, or I'll be glad to forward correspondence to him.

How the Computer Reads Music

Coded music, using the special music description language, is contained on paper tape. The tape is read into the computer, which controls the organ. Future plans include direct music-keyboard input on an organ that will enable it to double for input and performing. This, of course, eliminates the need to code music. Right now, however, his music description language is quite sophisticated. It includes capabilities to insert commands that permit the music to be slowed or speeded up, and commands for volume variations.

In Concert

Since audiences do not remain excited long when listening to recordings (with which computerized music may be equated), direct performer-composer communication with the musical keyboard has turned out to be the best course of action. That brings us full circle from hand-played music to completely computer-controlled music and back to manual keyboard control. This loop occurred during the course of three concerts at the All Saints Church in Pasadena. All three were performed on a 90-rank Schlicker Pipe Organ, using the PDP-8. The first concert featured computer-controlled notation, with Prentiss at the keyboard controlling the stops. Stops regulate the intensity of the music. Prentiss ran into some difficulty during one of the more complex pieces. He said, "Parts of the piece require that the stops be pulled out and put back in time. Sometimes I didn't get it just right and it tended to hurt the final effect." So he decided to computerize the stops as well, which were wired, as were the keys, to the computer. The program was expanded, and a great deal of time was spent programming the computer to most effectively and dramatically control the registration, or degree of intensity, of the music.

The first concert created an element of anxiety for the performers. Would it work? Would it all go smoothly? Prentiss and associates were tense. All went well, though, and they were relieved and moved by their success.

Preparation for the second concert took six months. It was completely automated, with an associate controlling the computer from backstage as Prentiss sat at the organ. It played itself entirely. Even pauses between pieces were programmed to allow sufficient time for applause and introduction to the next piece. The performers were well prepared, and the concert went off without a hitch. But the edge of excitement was lost, both for Prentiss and for the audience.

So Prentiss and the performers decided to bring the human element back into the picture for the third concert. As Prentiss puts it, "People can relate to the vulnerability and feel a degree of success along with the performer, as if they were pulling for him." The Teletype was moved onto the stage alongside the organ. That's where Prentiss sat. His measure of control was to hit the space bar to create a pause, either to sustain a note or chord, or to introduce the next number. Further control was added with the use of a

Texas Instruments Silent 700, offstage, which enabled Prentiss to sit at the organ console and, using a modem, effect real-time changes in the music programming.

During the 15 minutes or so it takes to "tune up" the computers or rewind tapes after testing the system, Prentiss is uncomfortably aware of his sole reliance on the Teletype and modem. At concert time, he is struck with a moment of uncertainty as he hits the space bar and wonders if it is going to do anything. He thought as time went on that it would become easier, but "It seems to get harder. There's much more hardware." And there are always ideas for last minute changes and improvements.

His fears are based on an experience one night before a concert, when nothing wanted to work. "We turned it on, and suddenly all the pipes were blasting at us. To this day I don't know why. After checking everything from cables to switches, I finally turned it off and went for a walk. When I came back and plugged everything in, it all worked fine."

Though Prentiss and his cohorts experience the same jitters that Broadway players do, he does not consider himself a performer. He sees his role primarily as one of delegating responsibility, much as an engineer does.

Record Albums

Prentiss and company have recorded two albums, both called "Unplayed by Human Hands." Prentiss feels the title implies "Played by Human Mind." The record jackets were illustrated using computerized art (what else?) by artist and friend Robert Earl Dewar. A sample of the titles indicates a wide range of music: Joplin's "Maple Leaf Rag," Rimsky-Korsakov's "Flight of the Bumble Bee" and "Fantasy: Everything is Computerized," co-composed by Prentiss Knowlton and Vladimir Ussachevsky. The last piece is interesting because it is humanly impossible to play; it was written to be played by computerized organ.

To my untrained, but appreciative ear, and to my mind, the music is wonderful, even astonishing. It's true music, dramatic and accurately performed. But I've asked professional musicians to listen to the albums. Now I'm sorry they knew in advance that the music was computerized as their reactions were certainly influenced by that foreknowledge. Rod Skelding, a professional pipe organist from Manchester, England, remarked that it was "soulless," though his respect for the accomplishment was not dimmed.

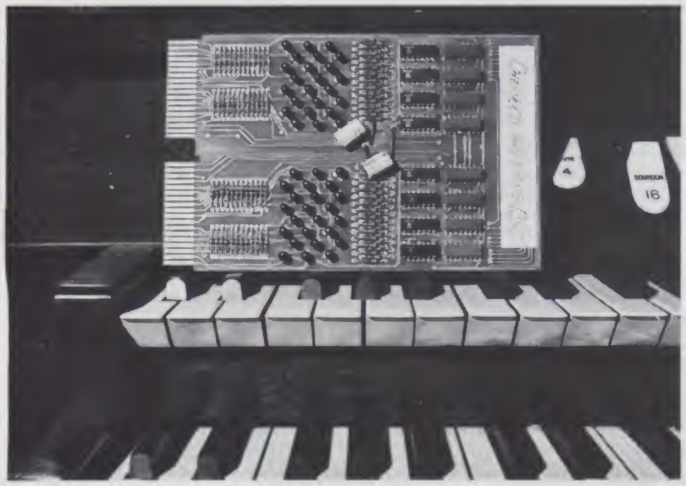
Dr. Knowlton has installed his own smaller pipe organ in Pasadena's Crown Theatre. It has been interfaced to the PDP-8 and revisions are under way to interface it to the Imsai. The manager of the Crown Theatre, Bruce Barkis, confesses he knows nothing about computers or programming. When asked how he had the courage to present computerized music, he said, "Our audiences are looking for a rare treat, and though organ technicians were appalled at connecting an organ to a computer, audiences are impressed enough to care little about how the music was achieved.

"If they don't know who's playing it, even a trained musician would have difficulty determining it is computerized. What tickles me is that such beautiful music is coming from something that looks like a common Teletype."

About Prentiss's ability, Mr. Barkis notes all the nuances of musical expression are in his music. "There are two kinds of musicians. One is a technician, and the other can play music with life, vigor and vitality. Prentiss can make a simple scale sound like a symphony. He gives his music color and expression. He's a true musician."

Who Benefits Most?

Besides computerized instruments being wonderful toys for the rich hobbyist, composers stand to benefit most from them. Now they might be spared the tedium of handwriting all notation and orchestrating music. They would also have a way to present original music to, for example, a conductor, on tape, to be heard as it was entered through the computer, and output through the organ and recorded on magnetic tape. No longer does a composer have to hope the conductor will pass music out to his orchestra in order to demonstrate the music. And complex rhythms, humanly impossible to play, might now be accomplished with computerized music.



This board handles four octaves, or 48 notes. A set of latches holds either a one or zero using ordinary chips. Each chip holds 6 bits and allows a key to be driven at ordinary TTL level, withstanding up to 30 volts. Two steps are involved: increasing the voltage and conducting current. This creates enough voltage and current to drive a single pipe. Turning the pipe off reverses the process. The interface handles 30 connectors and requires 15 boards like this one. Altogether, the system provides 720 separate functions; the 90-rank pipe organ uses about 600 functions.

The cable from the computer to the organ is 300-feet long, allowing the computer to be located in a building separate from the church. Its transmission cable and the signal are sent through a twisted pair of wires. A single bit (1 or 0) becomes a pair, in which one wire is low voltage and one is high voltage, called a differential line driver. Though noise can occur when data is transmitted this distance, the signal is sufficient for the receiver to differentiate between a 1 or 0.

What Does it all Mean?

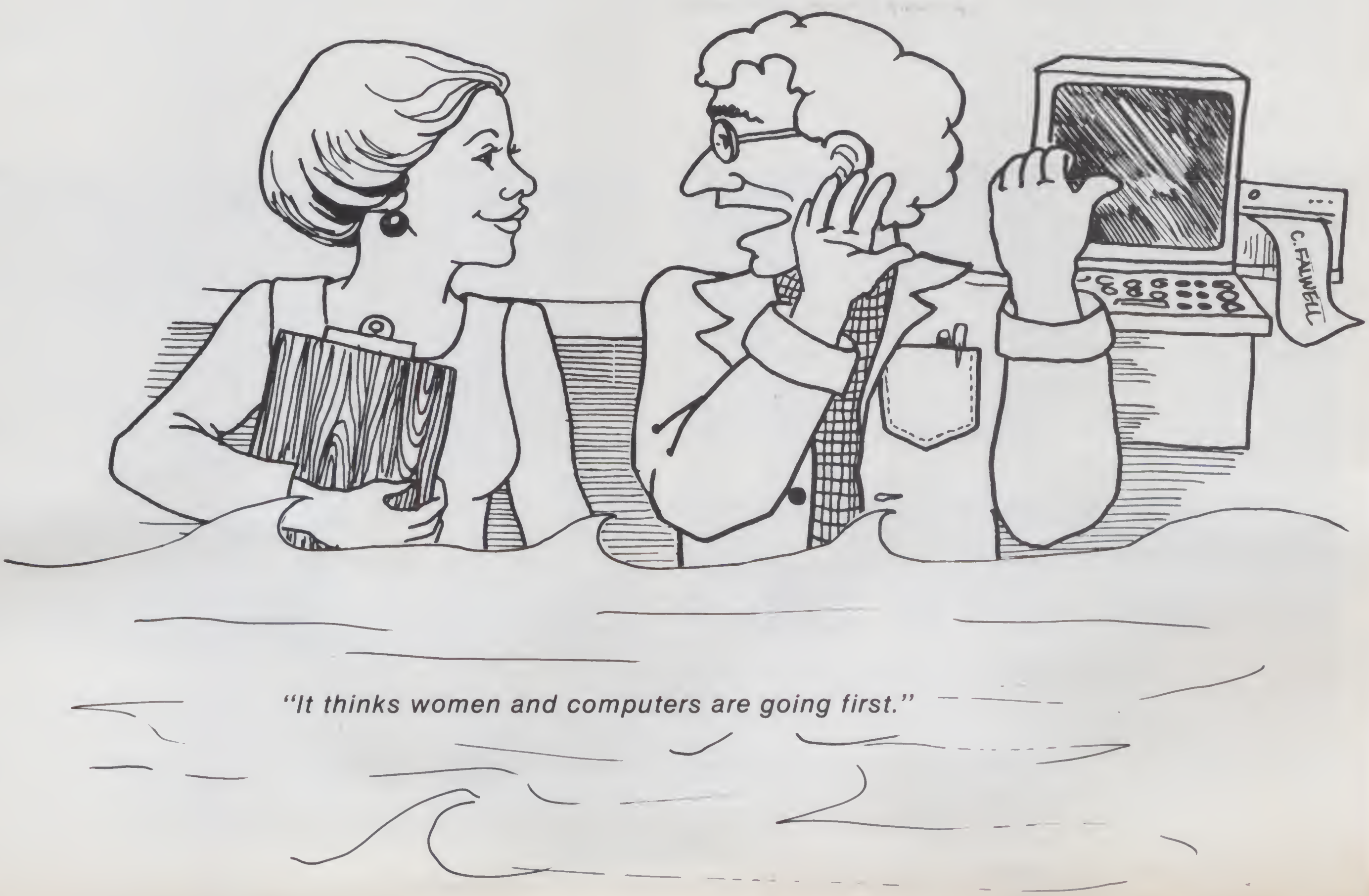
Artists seem to spend a lot of thoughtful energy reducing their achievements to philosophic terms. Prentiss Knowlton has several thoughts to offer for his. First, he sees computerized music as analogous to sculpture, chipping away at time rather than matter in space. Physical sculpture can only eliminate matter; music can add as well as subtract. The result, ideally, should last through time beyond the artist and his media because the information has been preserved in digital form.

Dr. Knowlton sums up his feelings about his achievement, saying, "If I can do something which improves man's ability to communicate with his own kind, arbitrarily at some distant point in the future, my life will be fulfilled."

If that sounds a bit corny, try to remember anyone of great accomplishment you've met or read about who was not driven by high emotion. ■

Something hobbyists can relate to . . . and duplicate. Prentiss's own home piano/organ is being built by a friend using LED photo transistor pairs, power supplies for LEDs and chips for TTL logic. It's designed for serial input and will run up to two megabits. A connector at the back will plug into the CPU.

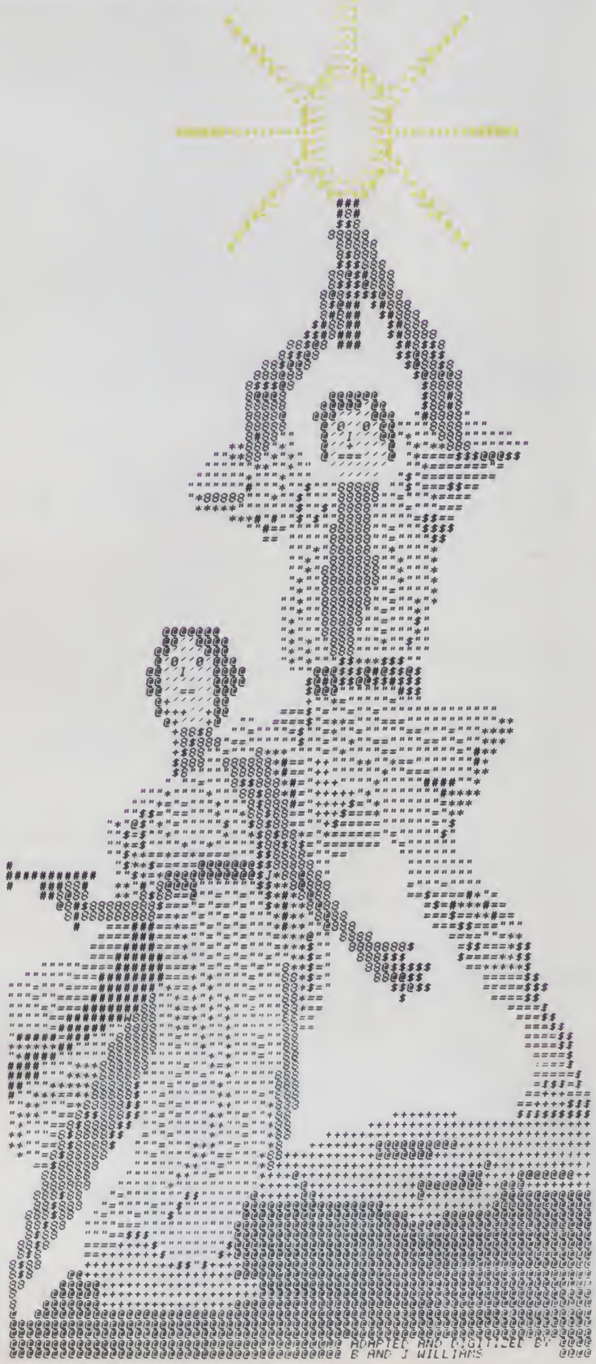
Since the keyboard contains 61 notes, 27 functions are left over for use to input data directly through the keyboard. Using the keys at the right end of the keyboard, the system can operate in four modes: direct input, start the computer running, repeat for instant replay, and the last defines the organ/computer relationship.



"It thinks women and computers are going first."


XITAN

Robert J. Bishop
1143 W. Badillo #E
Covina CA 91722



an exciting Apple-II graphics game

Last month's *Kilobaud* brought you "Rocket Pilot," the first in my series of arcade-type computer games. Now, here is my version of STAR WARS.

Actually, the program was written early in August on a Sunday afternoon. But, with the game about 90 percent completed, the power supply on my Apple-II suddenly rolled over and died, and the whole program evaporated. Unfortunately, I had not yet made a cassette tape of the game, and, as I had written the whole thing at the keyboard, I didn't even have a handwritten copy!

A couple of weeks later, with a new power supply, I began to reconstruct the program. Only this time I made sure to preserve updated states of the code on cassette tape every few minutes. The result of this effort is presented here.

The Game

In this version of STAR

Program A.

```
   0 REM *** STAR WARS ***
   1 REM COPYRIGHT 1977, R. J. BISHOP
   2 TV=1
   3 CLEAR=2217: SETUP=2429: DRAW=2304: UNDRAW=2308: PLOT=2048: TEST=2560: NOISE=2623: BELL=-198
   5 N=7
  10 GOSUB 5000
  20 CALL CLEAR
  25 GOSUB 1000
  30 POKE 15,0: CALL SETUP
  35 POKE 16,70: POKE 17,80: POKE 18,0
  40 GOSUB 3000
  50 CALL BELL
  60 POKE -16297,0: GR
  70 TIME=90*N: AMMO=25: HITS=0
  75 POKE 50,63
  80 VTAB 24: TAB 2: PRINT "       MAY THE FORCE BE WITH YOU!       ";: TAB 1
  85 POKE 50,255
  90 VTAB 22: PRINT "  TIME: ";(TIME+N-1)/N;: TAB 17: PRINT "SCORE: 0";: TAB 31: PRINT "AMMO: ";AMMO;
 100 X=X+( RND (21)-10)+(128- PDL (0))/3
 110 FOR K=1 TO 10: NEXT K
 120 Y=Y+( RND (31)-15)+(128- PDL (1))/2
 130 IF X<10 THEN X=10
 140 IF X>1309 THEN X=1309
 150 IF Y<10 THEN Y=10
 160 IF Y>1469 THEN Y=1469
 180 CALL UNDRAW
 200 POKE 16,X/10: POKE 17,Y/10
 210 CALL DRAW
 250 CALL TEST
 260 IF PEEK (19)=0 THEN 350
 270 AMMO=AMMO-1: TIME=TIME-1
 280 IF PEEK (19)#2 THEN 350
 300 GOSUB 2000
 310 GOSUB 3000
 320 HITS=HITS+1: TIME=TIME-N
 350 TIME=TIME-1
 400 TAB 9: PRINT "   ";: TAB 37: PRINT "  ";
 410 TAB 9: PRINT (TIME+N-1)/N;: TAB 24: PRINT HITS;: TAB 37: PRINT AMMO;
 450 IF TIME>0 AND AMMO THEN 100
 500 CALL BELL
 520 TEXT : CALL -936
 550 VTAB 7: TAB 11: PRINT "--- GAME OVER ---"
 570 VTAB 13: TAB 9: PRINT "YOU FIRED ";25-AMMO;" ROUNDS OF"
 580 TAB 11: PRINT "AMMO IN ";(90*N-TIME)/N;" SECONDS."
 590 PRINT : TAB 3: PRINT "YOU DESTROYED ";HITS;" ENEMY SPACESHIPS."
 600 PRINT : PRINT : TAB 5: PRINT "(HIT 'RETURN' TO PLAY AGAIN)"
 650 POKE -16368,0
 670 CHAR= PEEK (-16384)
 680 IF CHAR#141 THEN 670
 700 POKE -16368,0
 710 CALL UNDRAW: GOTO 40
1000 REM  DRAW GUNSIGHT
1010 FOR Z=0 TO 1
1020 POKE 18,Z
1030 FOR X=0 TO 139
1040 POKE 16,X
1050 POKE 17,0: CALL PLOT
1060 POKE 17,159: CALL PLOT
1070 NEXT X
1100 FOR Y=0 TO 159
1110 POKE 17,Y
1120 POKE 16,0: CALL PLOT
1130 POKE 16,139: CALL PLOT
1140 NEXT Y
1150 NEXT Z
1200 FOR X=45 TO 95
1205 IF X>=65 AND X<=75 THEN 1290
1210 POKE 16,X
1220 LONG=0
1230 IF (X-45) MOD 5=0 THEN LONG=3
1240 IF (X-45) MOD 10=0 THEN LONG=6
1250 IF X=45 OR X=95 THEN LONG=12
1260 FOR Y=80-LONG TO 80+LONG STEP TV
1270 POKE 17,Y: CALL PLOT
1280 NEXT Y
1290 NEXT X
1300 FOR Y=40 TO 120 STEP TV
1305 IF Y>=72 AND Y<=88 THEN 1390
1310 POKE 17,Y
1320 LONG=0
1330 IF (Y-40) MOD 8=0 THEN LONG=2
```

```
1340 IF (Y-40) MOD 16=0 THEN LONG=5
1350 IF Y=40 OR Y=120 THEN LONG=10
1360 FOR X=70-LONG TO 70+LONG
1370 POKE 16,X: CALL PLOT
1380 NEXT X
1390 NEXT Y
1400 RETURN
2000 REM  EXPLOSION SUBROUTINE
2010 CALL UNDRAW
2020 POKE 16,63: POKE 17,70
2040 POKE 15,29: CALL SETUP
2050 POKE 18,1: CALL DRAW
2055 CALL NOISE
2060 POKE 18,0: CALL DRAW
2065 CALL NOISE
2070 POKE 18,1: CALL UNDRAW
2075 CALL NOISE
2080 POKE 18,0: CALL UNDRAW
2085 CALL NOISE
2100 POKE 15,74: CALL SETUP
2110 POKE 18,1: CALL DRAW
2115 CALL NOISE
2120 POKE 18,0: CALL DRAW
2125 CALL NOISE
2130 POKE 18,1: CALL UNDRAW
2135 CALL NOISE
2140 POKE 18,0: CALL UNDRAW
2145 CALL NOISE
2500 POKE 16,X/10: POKE 17,Y/10: POKE 18,0
2530 POKE 15,0: CALL SETUP
2550 RETURN
3000 X=1290* RND (2)+10
3010 Y=1450* RND (2)+10
3080 CALL UNDRAW
3090 RETURN
5000 TEXT : CALL -936
5010 VTAB 7: TAB 10: PRINT "***  STAR  WARS  ***"
5020 VTAB 11: TAB 19: PRINT "BY"
5030 VTAB 13: TAB 12: PRINT "ROBERT J. BISHOP"
5040 VTAB 18: TAB 2: PRINT "(BASED ON THE MOVIE BY GEORGE LUCAS)"
5050 RETURN
```

0E00-	1C	09	0D	80	80	80	80	80
0E08-	80	88	80	9C	80	9C	80	FF
0E10-	80	9C	80	9C	80	88	80	80
0E18-	80	80	80	80	80	2C	0F	15
0E20-	40	04	20	18	30	10	19	10
0E28-	19	10	09	70	0D	F0	07	E0
0E30-	07	C2	0F	C8	3F	E8	7F	F8
0E38-	9F	F0	07	80	07	80	06	C0
0E40-	0C	E0	0C	60	18	30	10	10
0E48-	20	08	2C	0F	15	08	20	04
0E50-	40	04	40	84	40	42	80	62
0E58-	82	31	84	1B	EC	1E	78	0C
0E60-	38	04	30	04	20	04	20	04
0E68-	60	0E	F0	1F	B0	3B	98	23
0E70-	4C	41	42	02	20	02	20	00

Table 1.

WARS, you are one of the pilots of the X-wing fighters engaged in aerial combat with the Tie fighters of the Galactic Empire. Your mission is to destroy as many of the enemy spaceships as you can before time runs out.

The TV screen serves as your gun sight and displays a set of green cross hairs. The Tie fighters are purple and randomly enter your field of view from one of the four corners of the screen. As each fighter appears, you must maneuver it into your sight using the game paddle controls. When you're ready to shoot, push the RESET button on either paddle. If any part on an enemy ship is at the exact center of the cross hairs, it will explode and your score will increment by one. You have up to 25 rounds of ammunition, or 90 seconds, whichever expires first.

The Program

The STAR WARS game is written partly in BASIC (see Program A) and partly in machine language. An Apple-II with at least 16K of memory is required to run the program. Before entering the program, you must set LOMEM to 4096 and HIMEM to 8192. You are now ready to type in the BASIC portion of the code.

The machine-language part consists of three sets of subroutines (Subroutines 1, 2 and 3). The first set resides at 0800 to 08C4. The second set is from 0900 to 0992 and the third set is located at 0A00 to0A4D.

We're almost finished. The only thing left is to enter the table of hex numbers shown in Table 1.

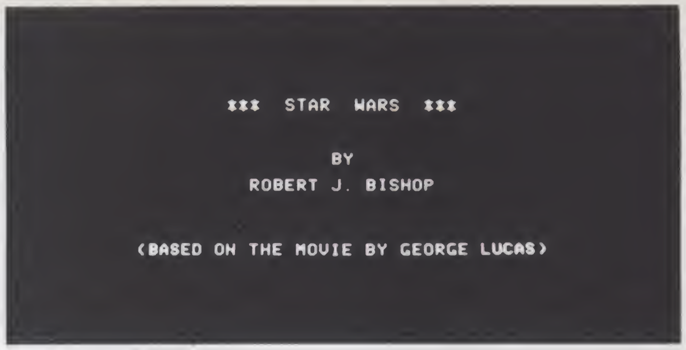
Now, go out and save the universe. And may the Force be with you.

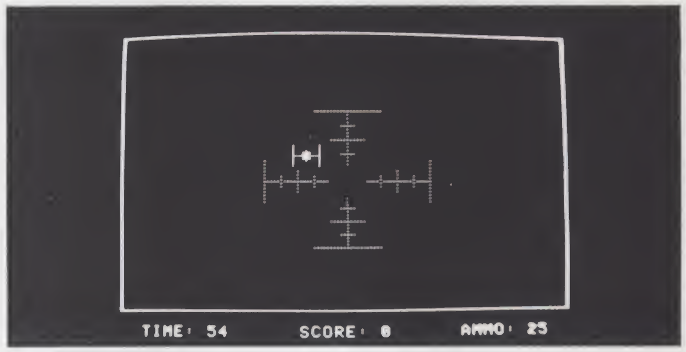
Cassette tapes of STAR WARS and other computer games are available from: Computer Playground, 6789 Westminster Ave., Westminster CA 92683.■

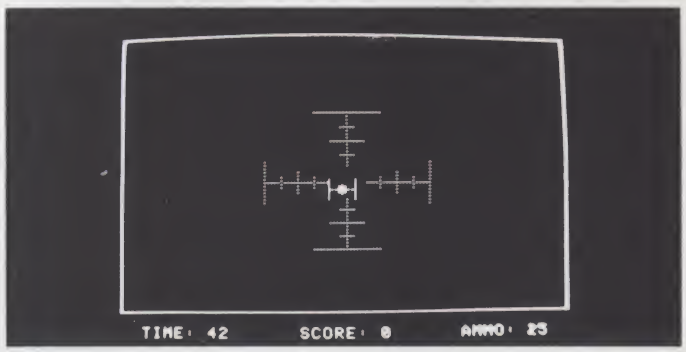
Subroutine 1.

```
0800-   98          TYA
0801-   48          PHA
0802-   20 26 08    JSR   $0826
0805-   11 14       ORA   ($14),Y
0807-   91 14       STA   ($14),Y
0809-   68          PLA
080A-   A8          TAY
080B-   60          RTS
080C-   98          TYA
080D-   48          PHA
080E-   20 26 08    JSR   $0826
0811-   49 FF       EOR   #$FF
0813-   31 14       AND   ($14),Y
0815-   91 14       STA   ($14),Y
0817-   68          PLA
0818-   A8          TAY
0819-   60          RTS
081A-   98          TYA
081B-   48          PHA
081C-   20 26 08    JSR   $0826
081F-   31 14       AND   ($14),Y
0821-   85 13       STA   $13
0823-   68          PLA
0824-   A8          TAY
0825-   60          RTS
0826-   20 47 08    JSR   $0847
0829-   20 6D 08    JSR   $086D
082C-   A5 14       LDA   $14
082E-   18          CLC
082F-   65 16       ADC   $16
0831-   90 02       BCC   $0835
0833-   E6 15       INC   $15
0835-   85 14       STA   $14
0837-   A4 17       LDY   $17
0839-   B9 40 08    LDA   $0840,Y
083C-   A0 00       LDY   #$00
```

```
083E-    60       RTS
083F-    EA       NOP
0840-    01 02    ORA   ($02,X)
0842-    04       ???
0843-    08       PHP
0844-    10 20    BPL   $0866
0846-    40       RTI
0847-    A5 11    LDA   $11
0849-    0A       ASL
084A-    0A       ASL
084B-    29 1C    AND   #$1C
084D-    85 15    STA   $15
084F-    A5 11    LDA   $11
0851-    6A       ROR
0852-    6A       ROR
0853-    6A       ROR
0854-    6A       ROR
0855-    29 03    AND   #$03
0857-    05 15    ORA   $15
0859-    09 20    ORA   #$20
085B-    85 15    STA   $15
085D-    A5 11    LDA   $11
085F-    6A       ROR
0860-    29 E0    AND   #$E0
0862-    85 14    STA   $14
0864-    6A       ROR
0865-    6A       ROR
0866-    29 18    AND   #$18
0868-    05 14    ORA   $14
086A-    85 14    STA   $14
086C-    60       RTS
086D-    A9 00    LDA   #$00
086F-    85 16    STA   $16
0871-    A9 E0    LDA   #$E0
0873-    85 18    STA   $18
0875-    A9 20    LDA   #$20
0877-    85 19    STA   $19
0879-    A5 10    LDA   $10
087B-    85 17    STA   $17
087D-    A0 06    LDY   #$06
087F-    A5 17    LDA   $17
0881-    C5 18    CMP   $18
0883-    90 0B    BCC   $0890
0885-    E5 18    SBC   $18
0887-    85 17    STA   $17
0889-    A5 16    LDA   $16
088B-    05 19    ORA   $19
088D-    85 16    STA   $16
088F-    18       CLC
0890-    66 18    ROR   $18
0892-    66 19    ROR   $19
0894-    88       DEY
0895-    D0 E8    BNE   $087F
0897-    06 16    ASL   $16
0899-    A5 17    LDA   $17
089B-    0A       ASL
089C-    05 12    ORA   $12
089E-    C9 07    CMP   #$07
08A0-    90 04    BCC   $08A6
08A2-    E9 07    SBC   #$07
08A4-    E6 16    INC   $16
08A6-    85 17    STA   $17
08A8-    60       RTS
08A9-    8A       TXA
08AA-    48       PHA
08AB-    98       TYA
08AC-    48       PHA
08AD-    A9 00    LDA   #$00
08AF-    85 14    STA   $14
08B1-    A2 20    LDX   #$20
08B3-    86 15    STX   $15
08B5-    A8       TAY
08B6-    91 14    STA   ($14),Y
08B8-    C8       INY
08B9-    D0 FB    BNE   $08B6
08BB-    E6 15    INC   $15
08BD-    CA       DEX
08BE-    D0 F6    BNE   $08B6
08C0-    68       PLA
08C1-    A8       TAY
08C2-    68       PLA
08C3-    AA       TAX
08C4-    60       RTS
08C5-    EA       NOP
08C6-    00       BRK
```





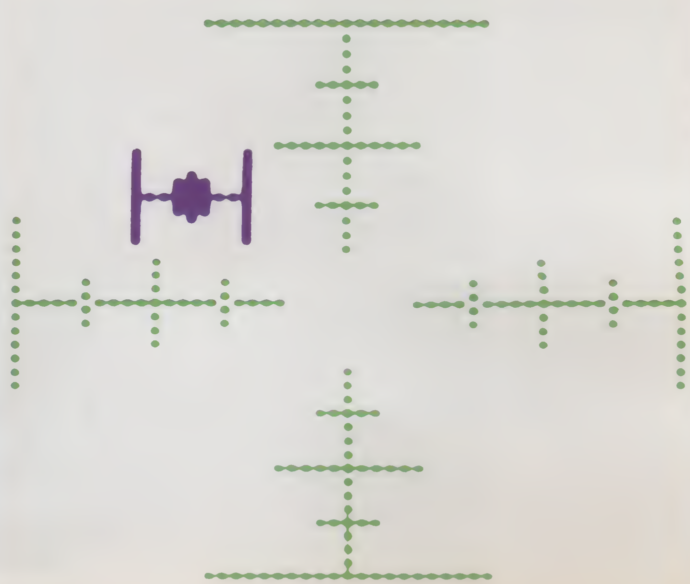


```
0900-   A9 FF       LDA   #$FF
0902-   D0 02       BNE   $0906
0904-   A9 00       LDA   #$00
0906-   85 1F       STA   $1F
0908-   8A          TXA
0909-   48          PHA
090A-   98          TYA
090B-   48          PHA
090C-   A5 11       LDA   $11
090E-   48          PHA
090F-   A2 00       LDX   #$00
0911-   AD 01 0F    LDA   $0F01
0914-   85 1B       STA   $1B
0916-   AD 00 0F    LDA   $0F00
0919-   85 1A       STA   $1A
091B-   20 26 08    JSR   $0826
091E-   85 1C       STA   $1C
0920-   A9 80       LDA   #$80
0922-   85 1D       STA   $1D
0924-   BD 02 0F    LDA   $0F02,X
0927-   25 1D       AND   $1D
0929-   F0 02       BEQ   $092D
092B-   A9 FF       LDA   #$FF
092D-   85 1E       STA   $1E
092F-   A5 1C       LDA   $1C
0931-   24 1F       BIT   $1F
0933-   70 09       BVS   $093E
0935-   25 1E       AND   $1E
0937-   49 FF       EOR   #$FF
0939-   31 14       AND   ($14),Y
093B-   50 0D       BVC   $094A
093D-   EA          NOP
093E-   31 14       AND   ($14),Y
0940-   05 13       ORA   $13
0942-   85 13       STA   $13
0944-   A5 1C       LDA   $1C
0946-   25 1E       AND   $1E
0948-   11 14       ORA   ($14),Y
094A-   91 14       STA   ($14),Y
094C-   A5 1C       LDA   $1C
094E-   0A          ASL
094F-   10 04       BPL   $0955
0951-   A9 02       LDA   #$02
0953-   D0 05       BNE   $095A
0955-   0A          ASL
0956-   10 03       BPL   $095B
0958-   A9 01       LDA   #$01
095A-   C8          INY
095B-   85 1C       STA   $1C
095D-   66 1D       ROR   $1D
095F-   90 09       BCC   $096A
0961-   66 1D       ROR   $1D
0963-   E8          INX
0964-   C6 1A       DEC   $1A
0966-   D0 BC       BNE   $0924
0968-   F0 05       BEQ   $096F
096A-   C6 1A       DEC   $1A
096C-   D0 B6       BNE   $0924
096E-   E8          INX
096F-   E6 11       INC   $11
0971-   C6 1B       DEC   $1B
0973-   D0 A1       BNE   $0916
0975-   68          PLA
0976-   85 11       STA   $11
0978-   68          PLA
0979-   A8          TAY
097A-   68          PLA
097B-   AA          TAX
097C-   60          RTS
097D-   A6 0F       LDX   $0F
097F-   BD 00 0E    LDA   $0E00,X
0982-   85 0E       STA   $0E
0984-   A0 FF       LDY   #$FF
0986-   E8          INX
0987-   BD 00 0E    LDA   $0E00,X
098A-   C8          INY
098B-   99 00 0F    STA   $0F00,Y
098E-   C6 0E       DEC   $0E
0990-   D0 F4       BNE   $0986
0992-   60          RTS
```

Subroutine 2.

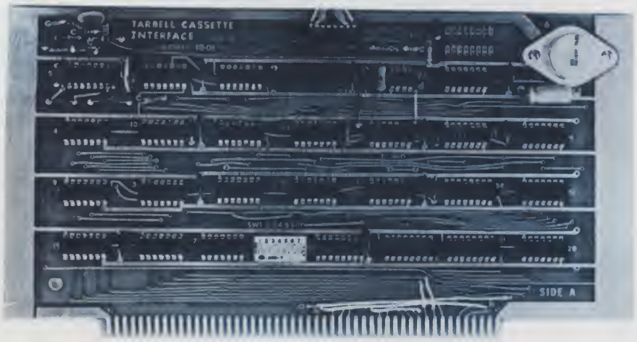
```
0A00-   A9 00       LDA   #$00
0A02-   85 13       STA   $13
0A04-   AD 61 C0    LDA   $C061
0A07-   0D 62 C0    ORA   $C062
0A0A-   30 01       BMI   $0A0D
0A0C-   60          RTS
0A0D-   A2 01       LDX   #$01
0A0F-   8A          TXA
0A10-   A8          TAY
0A11-   88          DEY
0A12-   D0 FD       BNE   $0A11
0A14-   AD 30 C0    LDA   $C030
0A17-   E8          INX
0A18-   E0 C0       CPX   #$C0
0A1A-   D0 F3       BNE   $0A0F
0A1C-   A5 10       LDA   $10
0A1E-   48          PHA
0A1F-   A5 11       LDA   $11
0A21-   48          PHA
0A22-   A9 46       LDA   #$46
0A24-   85 10       STA   $10
0A26-   A9 50       LDA   #$50
0A28-   85 11       STA   $11
0A2A-   20 1A 08    JSR   $081A
0A2D-   38          SEC
0A2E-   A5 13       LDA   $13
0A30-   F0 02       BEQ   $0A34
0A32-   A9 01       LDA   #$01
0A34-   69 00       ADC   #$00
0A36-   85 13       STA   $13
0A38-   68          PLA
0A39-   85 11       STA   $11
0A3B-   68          PLA
0A3C-   85 10       STA   $10
0A3E-   60          RTS
0A3F-   A0 40       LDY   #$40
0A41-   AD 30 C0    LDA   $C030
0A44-   BE 00 E0    LDX   $E000,Y
0A47-   CA          DEX
0A48-   D0 FD       BNE   $0A47
0A4A-   88          DEY
0A4B-   D0 F4       BNE   $0A41
0A4D-   60          RTS
0A4E-   EA          NOP
0A4F-   EA          NOP
0A50-   EA          NOP
0A51-   EA          NOP
0A52-   EA          NOP
0A53-   EA          NOP
0A54-   EA          NOP
0A55-   EA          NOP
```

Subroutine 3.



TARBELL SETS STANDARDS
For Hobbyists and Systems Developers

Sales to thousands of hobbyists over the past two years have proven the Tarbell Cassette Interface to be a microcomputer industry standard. Tarbell Electronics continues research and development to produce new and efficient components to fill hobbyists' changing needs.



TARBELL CASSETTE INTERFACE

- Plugs directly into your IMSAI or ALTAIR*
- Fastest transfer rate: 187 (standard) to 540 bytes/second
- Extremely Reliable—Phase encoded (self-clocking)
- 4 Extra Status Lines, 4 Extra Control Lines
- 37-page manual included
- Device Code Selectable by DIP-switch
- Capable of Generating Kansas City tapes also
- No modification required on audio cassette recorder
- Complete kit $120, Assembled $175, Manual $4

TARBELL FLOPPY DISC INTERFACE

- Plugs directly into your IMSAI or ALTAIR* and handles up to 4 standard single drives in daisy-chain.
- Operates at standard 250K bits per second on normal disc format capacity of 243K bytes.
- Works with modified CP/M Operating System and BASIC-E Compiler.
- Hardware includes 4 extra IC slots, built-in phantom bootstrap and on-board crystal clock. Uses WD 1771 LSI Chip.
- 6-month warranty and extensive documentation.
- PRICE:
  Kit $190 ...... Assembled $265

Compatible Disc Drives

Ask about our disc drives priced as low as $525.

TARBELL PROTOTYPE BOARD
Model 1010

- Gold plated edge pins
- Takes 33 14-pin ICs or
- Mix 40-pin, 18-pin, 16-pin and 14-pin ICs
- Location for 5 volt regulator
- Suitable for solder and wire wrap
- ALTAIR/IMSAI compatible

Price: $28.00

For fast, off the shelf delivery, all Tarbell Electronics products may be purchased from computer store dealers across the country. Or write Tarbell Electronics direct for complete information.

*ALTAIR is a trademark/tradename of MITS, Inc.

Tarbell Electronics
20620 South Leapwood Avenue, Suite P
Carson, California 90746
(213) 538-4251

T11

Tuned for Speed

hot-rod mods for your SWTP system

Don Gille
3628 Derbyshire Rd.
W. Lafayette IN 47906

My many years before becoming a computer hobbyist were spent racing cars at the local drag strip, the objective being to have the fastest car in the country (or world, depending on your ego). The need for speed has carried over to my computer system.

On the subject of computers, I purchased a Southwest Technical machine. Many of my friends have asked me why I bought that particular computer. Before you ask, here are some of the features I looked for in my future system. 1. I had $500.00 to spend. 2. I wanted a system with BASIC. 3. The system should have good hardware and software support. 4. The processor should be a popular one (6800, 8080, Z-80, 6502).

Sine wave generated by BASIC running very, very fast!

With these requirements in mind I jumped into the trade magazines and looked for the machine of my dreams. The systems available were very impressive but cost much more than I really wanted to pay. I couldn't help thinking about all the things I would be able to do with my fast computer (wrong!). I wanted a fast processor. After all I couldn't have any of my friends executing their programs faster! As I looked over the choices I noticed a 6800 system with BASIC which cost $400. All that remained was to send my money and wait for the computer.

Well, I received the kit, spent hours in assembly and

ended up with a machine that was not only slow but used a parallel port for serial I/O! After many hours of deliberation I researched the origin of the serial/parallel port. My first thought was that the factory had an excess of parallel ports on hand, and found they could be used for Teletype ports. Another thought: In order to make the system more impressive looking they put the larger parallel port on the board. An engineer's need to know led me to investigate the real reason for the parallel port, which is discussed in a moment. After deciding that I had purchased a "led sled" (drag strip lingo for a slow car), I looked for a way to make it faster.

Looking for a Low E. T.

On the drag strip the goal of the driver is to turn in a low elapsed time (E.T.). Simply, he who getteth to the finish line first winneth. The goal with a computer is the same; to have the fastest execution time of the competition (also called E.T.).

In my search for throughput ($10 word for gettin' it done) I discovered that there was indeed some thinking behind the mystery of the PIA (Parallel Interface Adapter) that thought it was an ACIA (Asynchronous Communications Interface Adapter). The story starts with the creator, Motorola.

In the Beginning . . .

Motorola, in their infinite wisdom, used the PIA to send serial data to a terminal connected to their evaluation board. This was done to allow a paper tape punch/reader to be controlled by the other free lines of the PIA. Good idea! It was only a matter of time until the good idea became a pain.

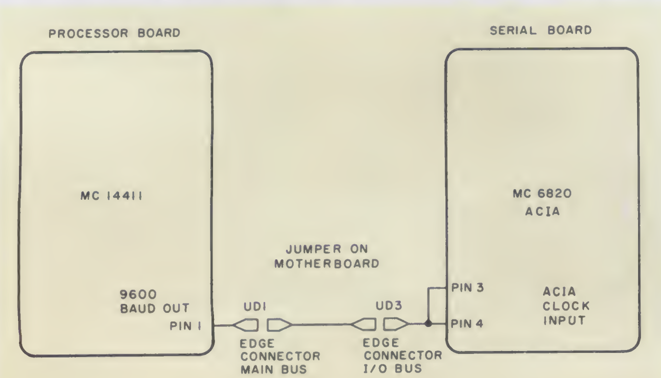
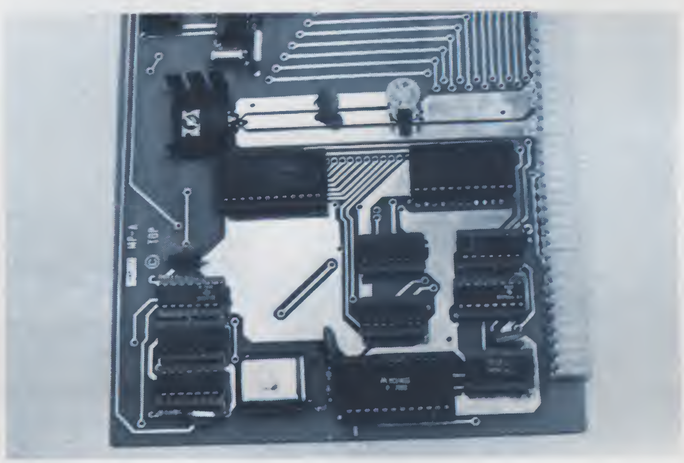
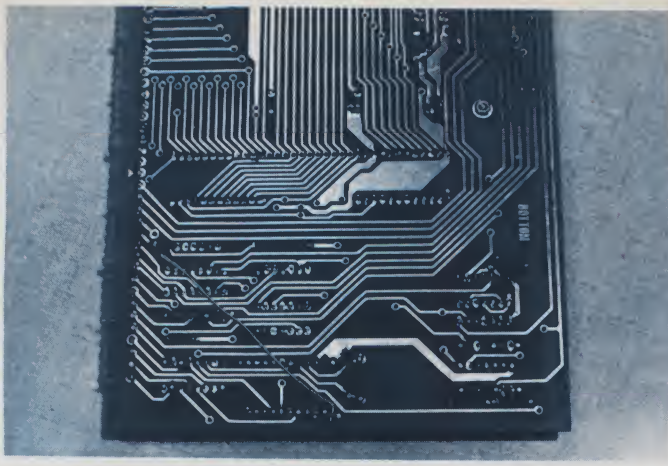


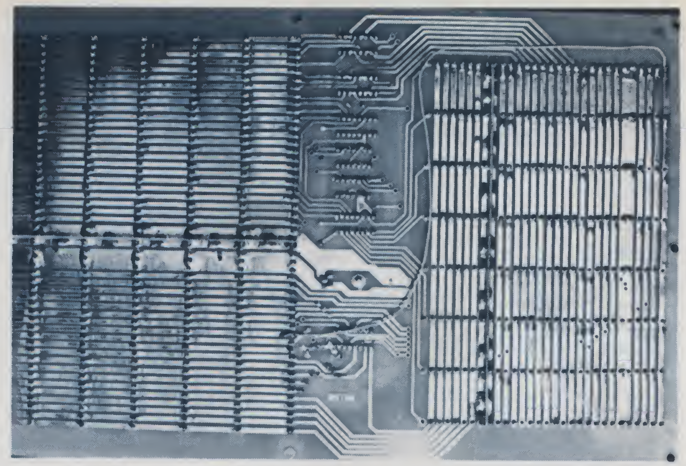
Fig. 1. Block diagram.



Top of the processor board showing the MC14411 IC (lower chip in photo).



Bottom of the processor board showing connection to UD1 and pin 1 of the MC14411.



Bottom of the motherboard showing the jumper from UD3 to UD1.

Having spent loads of money on the development of this interface, Motorola decided to leave well enough alone. They had developed Mikbug, the most powerful monitor of its time, to drive the strange interface. Mikbug was the first real step toward making the low cost hobbyist computer a reality, even though it was developed for industry.

You can still buy the Motorola evaluation board with the serial PIA, and it's a good low-cost system for the budget hobbyist (aren't we all). It will give hours of enjoyment for very little money.

But the question still remains; why did SWTP use the PIA instead of the ACIA?

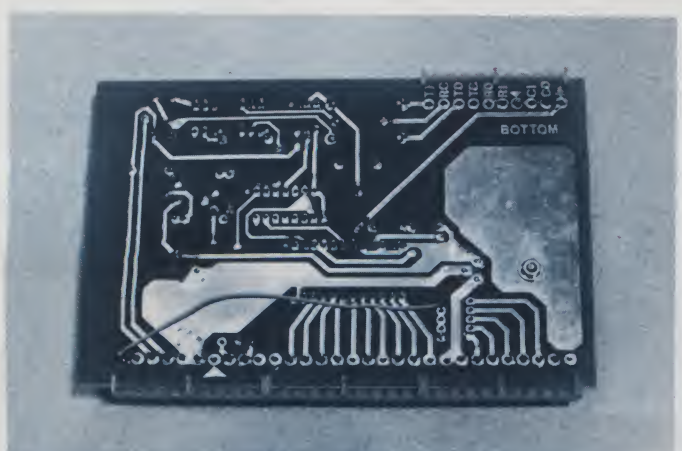
Follow the Leader

The story continues in an electronic kit manufacturer's factory out West where it's hot. Here the engineers no doubt used the PIA in their design to avoid replacing Mikbug.

The company, SWTP, designed their system to use the PIA for the interface to your Teletype (what's a video terminal?). The idea was sound, knowing that the creator had such good luck with it.

Today, however, the story is different. The hobbyist market has become more advanced, and the terminals used by most hobbyists are faster. This has a marked effect on the throughput of a computer system. The use of this PIA/serial interface on your system in the battle for speed is like having 1000 pounds of cement in the trunk of your dragster!

Unlike SWTP, most manufacturers use a strange device called a UART for parallel to serial conversion.



Bottom of the serial I/O board showing connection between UD3 and the normal baud rate select pads. Place card in slot 3 of the SWTP I/O bus for these mods.

The Strange UART Device

If you thought UART was a branch of the FBI, you're wrong. A UART is an asynchronous communications receiver transmitter. It uses a register to hold up to 11 bits of information, which it turns into a serial data stream (much like a shift register). This data stream is sent over one wire to the terminal, and over another back to the computer. The small number of wires used for transmission is the main advantage of this type of interface.

The computer uses a UART to send serial information, and the terminal also uses one to reverse the process and recover the parallel information. The terminal's UART stacks the serial data in its register for display on the screen. The same is true for the receiving section of the processor's UART, only this time the data is placed in the computer's memory.

You can see that in order to use a PIA for serial communication (in place of a UART) the parallel to serial conversion of the data must be done with software. This is the case with Mikbug (arg!). The software takes in data in parallel form and sends it out a bit at a time to one line of the control interface. This process takes up loads of time, losing the race for you! So, in answer to the cry for speed, here are the secrets to turn that clunker into a high-power beast!

A Patchwork Processor

This mod uses the serial (ACIA) port of the machine (SWTP only) for I/O. Other authors have shown how to urge the PIA to higher speeds (1200 baud). But still the fact remains; the processor must do the work of data conversion. With this mod you will be using the control interface to load your programs but execution will be through the serial (ACIA) port at speeds up to 9600 baud! This will allow your favorite BASIC or assembly language programs to execute faster (even at 1200). You now will be able to weed through long programs in much less time at 9600 baud.

On the drag strip larger tires and dual carburetors are added to get extra horsepower. In the SWTP you don't add anything to the basic system. All the equipment needed to execute and display programs is in the machine — unused!

To make the mods your machine need only be down a short time, about the same as

REX™ IS HERE
AT A COMPUTER STORE NEAR YOU!

We took the most desirable small computer features and put them into one machine – for $2495 (or less)!

REX™, RCC's new personal computer, is space-age modern in appearance, regal in features, but down-to-earth in price.

REGAL FEATURES

Just add one of our video monitors (or our serial channel and your hard copy terminal) and FORTRAN IV to the basic **REX** system, and begin developing your application programs.

REX includes an S-100 motherboard with five card slots, 24K of RAM, Z-80 CPU, video display interface, bootstrap PROM, two direct memory access channels, and powerfail and vectored interrupt circuitry.

AND THAT'S NOT ALL!

REX also includes a modern walnut-panel cabinet, ASR 33-type keyboard, 15 amp power supply, microfloppy drive (double-sided recording) and interface (including 8 bit parallel I/O port). All standard.

SOFTWARE TO PLEASE A KING

REX includes a powerful system Monitor and a Floppy Disk Operating System/File Management System as standard features. Low cost software options include ANSI standard FORTRAN IV, Extended Disk BASIC, Text Editor, and Linking Loader.

A KING'S CHOICE OF OPTIONS

Black and White or Color Video Monitors ■ Programmable CRT Character Generator ■ Expansion Microfloppy Drives ■ RAM Memory to 64K (does not require bus slots) ■ 120 CPS 80/96 Column Printer ■ RF Modulator and Color TV Driver for Home Sets ■ Programmable Serial Channel ■ Data Communications Modem.

ROYAL QUALITY AT A SURPRISINGLY LOW PRICE

The **REX** System is shipped assembled and tested for **$2495**, list price. See your computer dealer for his special low introductory price (for short time only).

DEALERS

CALIFORNIA
San Francisco . . . . Computer Store of San Francisco
San Jose . . . . Byte Shop III
Santa Clara. . . . . Byte Shop-Computer Store

COLORADO
Englewood. . . . . Byte Shop, Inc.
Boulder. . . . . The Byte Shop

IOWA
Davenport . . . . Computer Store of Davenport

MICHIGAN
Royal Oak . . . . Computer Mart of Royal Oak

MINNESOTA
Minneapolis . . . . .Computer Depot

NEBRASKA
Lincoln. . . . . .Microtech
Omaha . . . . .Omaha Computer Store

NEW YORK
New York City . . . . .Computer Mart of New York, Inc.

WISCONSIN
Beloit. . . . . Austin Computers

rcc™ **realistic controls corporation**

404 West 35th Street Davenport, IA 52806
Toll Free (800) 553-1863 In Iowa (319) 386-4400

a fast pit stop. Your pit should contain a hot soldering iron and pliers (no gas or oil is needed). Now dive in!

First, see that the machine is unplugged, as your burnt body on the floor might upset passersby. Take out all the cards and use conductive foam to protect the MOS chips (6800, 6820, 6850, etc). You could also use aluminum foil for this. Second, locate the MC 14411 baud rate generator on the processor board and flip the board over to inspect the bottom side.

Note which pin on this chip is number 1; we will be connecting to this.

Now, locate the pin-outs on the edge connector called UD1 and UD2. Carefully solder a wire to the number 1 pad (making sure not to short other pins around it) and extend it to UD1 or UD2, whichever is not used. Be sure you haven't used these lines for anything else as damage will result!

Flip the motherboard over and locate the pin (UD1-UD2) you selected on the processor board. You should remove the 14411, solder a wire on this pin and run it to an unused pin on the I/O bus along the rear of the motherboard). On the I/O bus you should use the UD3 or UD4 line provided they are not in use already. You now should have the main bus connected to the I/O bus and pin 1 of the MC 14411 connected to the edge connector of the processor board.

Now find the serial board amidst the rubble, and locate the baud rate select jumper on this board. Remove the select wire from this jumper area. Now flip the board over and locate these same jumper pads on the back. Solder a wire from the pads which lead to pins 3 and 4 of the MC 6850 ACIA to the edge connector pin you used under the motherboard on the I/O bus. Do not connect anything to the other pads – or else!

To finish the job check your work, put all the boards back in and check your work again (many a motor has scattered due to haste!). When you power up, the machine will be able to run at 9600 baud. To return to the slower speeds (heaven forbid!), connect the wire you ran to the edge connector to a baud rate select switch as described in *Kilobaud* #5 ("Speed Up Your 6800," p. 49).

To complete the pit stop, the software needs some modification. As you know, the machine will normally come up on the control interface, not the serial. So we must add a routine to replace the output and input instructions in any program you wish to run more quickly. You may wish to run assembly language programs. Or, if you can find the I/O routines in your favorite BASIC, it too will spend less time out to lunch and more time blowing the keys off of the competition!

The I/O routines in Mikbug are located at E1AC and E1D1 hex. E1D1 is used to output the ASCII character in the A accumulator. Change this to A020 hex. Now look for E1AC and substitute A014 hex. This should allow you to run at full speed through the serial port. The program which sets up and drives the ports is included with some hints on using a very good BASIC through the serial port. When playing games such as Startrek where the screen needs constant updating this will reduce the "ho hum" waiting period to a blink (try Chase!, also found in an earlier *Kilobaud*).

Finally, set your terminal to 9600 baud, one stop bit and upper case. Then invite your doubting 8080 friends over with their machines.

Making Your Friends Mumble Under Their Breath

I had heard too many jokes about my slow $500 system from another dollar-conscious owner, so I challenged him to a match race.

The new, more powerful

To run Mits BASIC on your SWTP use the following patches.

Address	Hexidecimal
0004	7EA0
0006	30
0417	A020
0420	A014
0422	01
0618	B680
061A	0C47
061C	2501
061E	39
061F	B680
0621	0D84
0623	7F01
0625	0101
0627	0101
0629	0101
08A8	A020

To use the serial port on other programs use the following code.

A014	B680	
A016	0C47	
A018	24FA	
A01A	B680	
A01C	0D84	
A01E	7F39	
A020	36B6	
A022	800C	
A024	4747	
A026	24F9	
A028	32B7	
A02A	800D	
A02C	39	
A030	8603	
A032	B780	
A034	0C86	*Initializes*
A036	15B7	*ACIA*
A038	800C	
A03A	39	

Enter these patches through the control interface to your program and it will come up on the serial port your card should be in slot number 3. Now you're TUNED FOR SPEED!

For use in BASIC A03A = 7E 18F9.

For use in other programs A03A = 39.

Program A. Software patches to use the serial port.

processor was ready, and we awaited the first runoff. The place: Benchmark Raceway, Purdue University — home of the fastest (or sometimes the slowest) computers in the world. The opponent: an Altair 8080. We were off . . .

At the finish it was the 6800 by 15 seconds (out of approximately two minutes average) over the out-classed Altair! Note: This was an actual benchmark run on the SWTP 6800 and an Altair 8800. The reason the Mits machine was so slow is that its clock speed is about half that of the Southwest Tech, which can be cranked up even more (maybe the subject of another article).

After you prove the speed of your 6800 to the doubting 8080 crowd and stop their insults, tell them the moral of the story: The man who throws mud loses ground! Gentlemen, start your 6800s! ■

Carl D. Warren II
2980 West 235th Apt 12
Torrance CA 90505

You'll Be Tickled by Fickled

a charting and diagramming aid

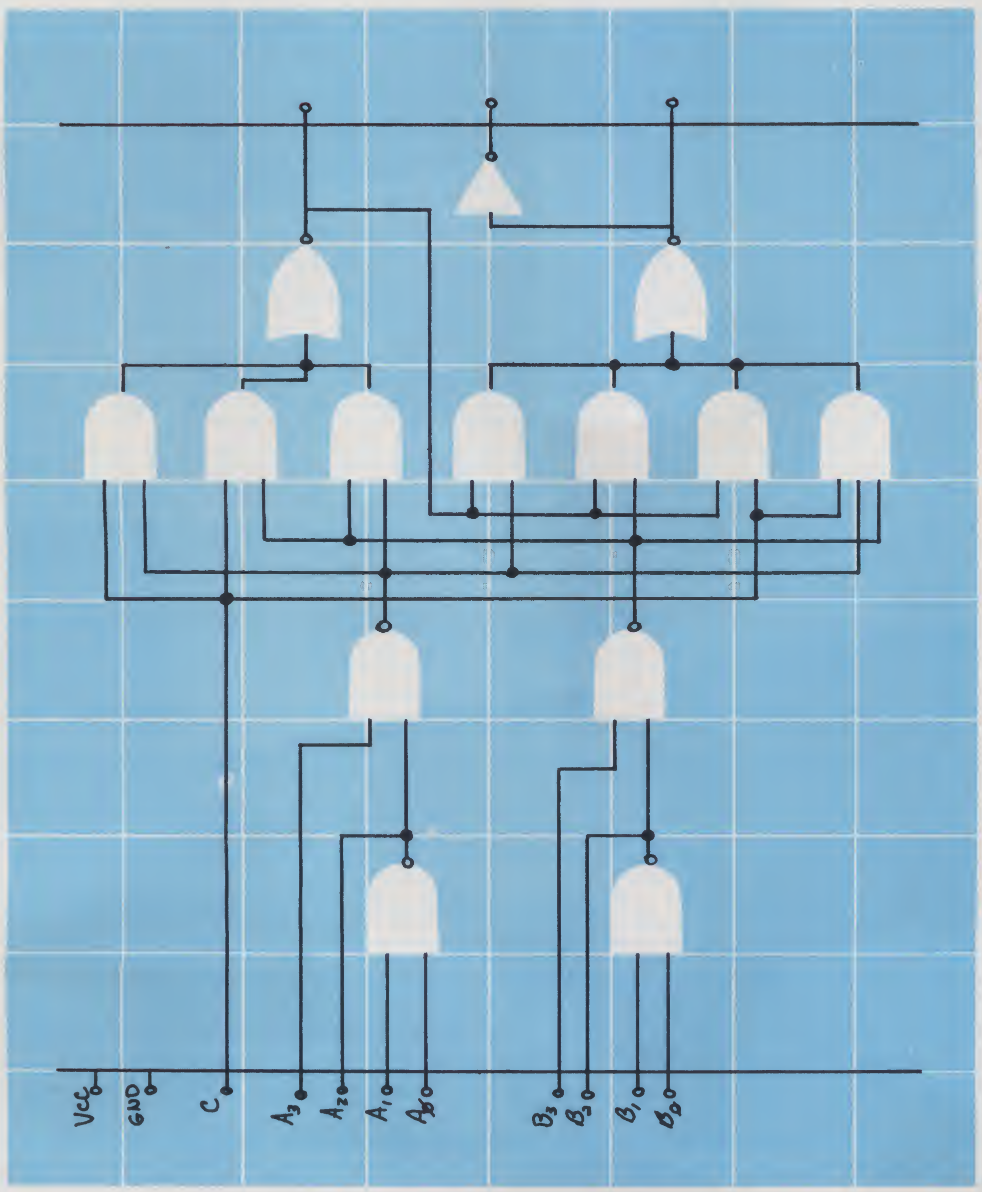


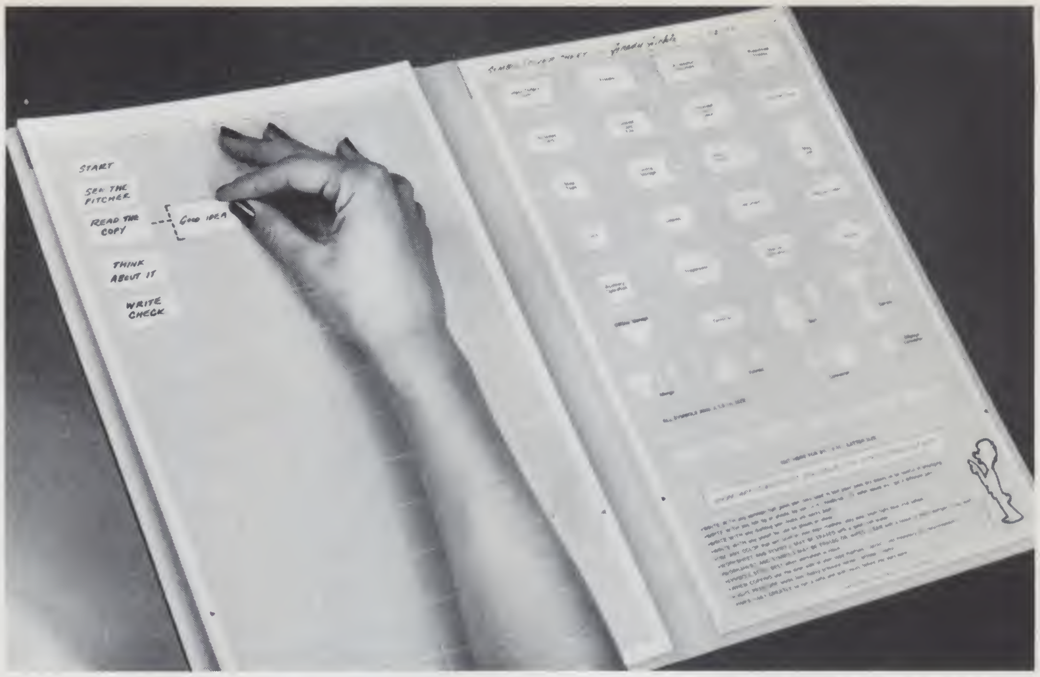
Fig. 1. Example of a logic diagram from a copied Fickled document.

How often have you begun a project that required either software or hardware design and wished there was a more efficient way to draw the flowchart or logic diagram? Probably as often as I have, which is every time I work on something. Well, all is not lost. An Orange, California, company called Fickled Thinking Aids has developed a low-cost, reusable system for flowcharting, logic diagramming or just doing your own thing.

The Fickled System

The Fickled system is similar to the vinyl stick-on toys that children play with and consists of soft vinyl ANSI (American National Standards Institute) computer flowcharting and logic symbols on precut padded sheets. Also included are work sheets with one-inch grid ruling and title block to assist in your layout. The symbol pads have cover sheets that indentify each symbol — great for the novice. The entire system is enclosed in a high-quality padded vinyl folder, which serves as a working base and provides portability.

The Fickled system uses the simple principle of electrostatics to hold the symbols on the work sheets. Therefore, no glue or other adhesive is required, and the



symbols are reusable since you need only peel them off the work sheet and replace them on the symbol pads.

The system is designed so you can write on both the symbols and work sheets, to aid in the depiction of your idea. Either ball-point pen or pens similiar to a Pentel can be used. However, the Fickled folks do offer a special pen for a dollar that works extremely well. In any case, the writing is easily wiped off the symbols or work sheets with a cloth or the special eraser that is available for 50 cents.

The system is flexible enough to allow for just about any presentation you have a need for. A good example is doing a hardware or software article for *Kilobaud*. Using the Fickled system, you can send in a clean, unsmudged copy of your work, because the system is designed with a field density to permit a good print from nearly all copy machines. Fig. 1 is an example of a Fickled document.

Summary

Fickled Thinking Aids let you do design work as fast as you can think. When you use the system you can make changes as fast and often as you want without making a mess. The system is inexpensive — $8.95 for the starter set shown in this article. For more information, write to: Ron Ford, Fickled Thinking Aids, P.O. Box 6064, 990-M Enterprise St., Orange CA 92667. ■

Sgt. Kenneth E. Wigger
125 North 1st St.
Wood River IL 62095

Ready on the Firing Line?

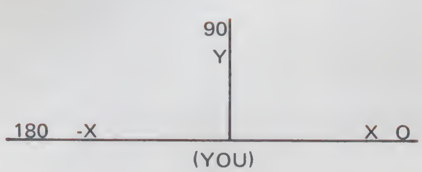
"Artillery Practice" for the SR-56

After reading Herman DeMonstoy's article, "Artillery Practice," in *Kilobaud* (June 77), I wondered if possibly it could be adapted to a small programmable calculator. Lacking 8K BASIC and 10K of memory, I admit that adaptations of computer programs for programmable calculators can be very trying. On several occasions, I have found that I have been able to adapt a good program only to find that the 100-step memory was overrun by 30 or 40 steps. For those who are unaware of a programmable calculator user's definition of frustration, I will relate it to you here.

Frustration is successfully adapting a program only to find yourself with a 101-step program and a 100-step memory. In the true tradition of Murphy's Law, you innocently assume that all you have to do is go back over your program and throw out an unnecessary step. After your first few confident minutes, you slowly realize that every last one of those steps was needed. Still somewhat composed, you accept that you might have to redo the program a little. Forty sheets of paper later, you find that you have a longer program than you started with and cannot find your original program. At this point, your mind doesn't make sense out of things like CMs, EXC, int, bst or subr anyhow. My advice is to give up and try tomorrow or the next day. I am not going to admit how many sheets of paper I used to adapt this program, or why my old analytic geometry book from college is still under my pillow.

The instructions for the game are typed out on a sheet of paper since only a few letters really look like the symbols on my Texas Instruments SR-56.

I would like to give credit to Peter Stark for the ran-



Example 1.

Example 2.

dom-number-generator subroutine ("Submarine!" *Kilobaud*, Feb. 77), which I used in my adaptation of DeMonstoy's "Artillery Practice." Stark's subroutine enabled me to program games that require random numbers without unnecessarily using up my very limited program step memory.

Basic game instructions and an SR-56 user's-instructions form are included to facilitate programming.

Game Instructions

Artillery Practice is a computer simulation game that simulates range-firing of artillery. The simulated range in which you will be firing is shown in Example 1.

You will be given only the X and Y coordinates of your target. You must determine the proper azimuth and elevation angle needed to hit it. The maximum range of your artillery piece is 4648.9 feet, and this is only possible if you fire at an elevation angle of 45 degrees. Any elevation angle above or below 45 degrees will reduce proportionally the range of your weapon. After you fire your artillery piece, the distance by which you missed will be displayed. Now that you understand the game, go to the SR-56 user's instructions and start playing. The trajectory is shown in Example 2. ■

Step	Procedure	Enter	Press				Display
1	Prepare to store data in registers.	N/A	CLR	RST	2nd	CMs	0
2	Store initial data to set up game for play.	385.7	STO	0			385.7
		.00001	STO	4			.00001
	(Insert initial random number seed between 0 and 1)	.9	STO	5			.9
		2600	STO	6			2600
	(Example game)						
1	Start game and display initial X of target.	N/A	R/S				-2139.0 (X)
2	Display Y of target.	N/A	X → T				974.4 (Y)
3	Enter elevation angle of shot.	23	X → T				-2139.0
4	Enter azimuth of shot.	160					160
5	Fire and find out distance by which you missed.	N/A	R/S				1017.6
1	Display new X of target.	N/A	R/S				719.5 (X)
2	Display new Y of target.	N/A	X → T				1512.1 (Y)
3	Enter elevation angle of shot.	10	X → T				719.5
4	Enter azimuth of shot.	75					75
5	Fire and find out distance by which you missed	N/A	R/S				308.9
	Each new play is continued in this same fashion.						

User's instructions.

Loc.	Code	Key
00	49	* fix
01	01	1
02	57	* subr
03	07	7
04	04	4
05	33	STO
06	01	1
07	92	•
08	05	5
09	32	X → T
10	57	* subr
11	07	7
12	04	4
13	34	RCL
14	05	5
15	17	* INV
16	47	X ≥ T
17	02	2
18	02	2
19	01	1
20	93	±
21	64	X
22	34	RCL
23	07	7
24	94	=
25	33	STO
26	02	2
27	34	RCL
28	01	1
29	32	X → T
30	34	RCL
31	02	2
32	41	R/S
33	33	STO
34	03	3
35	32	X → T
36	64	X
37	02	2
38	94	=
39	23	sin
40	64	X
41	34	RCL
42	00	0
43	43	X^2
44	54	/
45	03	3
46	02	2
47	94	=
48	32	X → T
49	34	RCL
50	03	3
51	26	*F(n)
52	02	P → R
53	52	(
54	34	RCL
55	09	9
56	74	—
57	34	RCL
58	02	2
59	53	)
60	43	X^2
61	84	+
62	52	(
63	34	RCL
64	08	8
65	74	—
66	34	RCL
67	01	1
68	53	)
69	43	X^2
70	94	=
71	48	* $\sqrt{X}$
72	41	R/S
73	42	RST
74	34	RCL
75	05	5
76	64	X
77	07	7
78	45	Y^X
79	09	9
80	64	X
81	34	RCL
82	04	4
83	94	=
84	17	* INV
85	29	int
86	33	STO
87	05	5
88	64	X
89	34	RCL
90	06	6
91	84	+
92	05	5
93	00	0
94	00	0
95	94	=
96	33	STO
97	07	7
98	58	* rtn
99		

Registers

0	385.7
1	Y of target
2	X of target
3	Azimuth
4	.00001
5	Random #
6	2600
7	Random coordinate
8	Y of shot
9	X of shot

Notes

* denotes 2nd

Program listing.

John Blankenship
DeVry Institute of Technology
828 W. Peachtree N.W.
Atlanta GA 30308

Expand Your KIM!

Part 3: bus control board and memory

By now, I hope many of you have completed construction of the KIM-1 System cabinet, mainframe and power supply covered in my last article. Now, you will see how easy it is to connect many Altair-compatible peripherals to a 6502 bus structure.

Let's begin by looking at the KIM bus and timing system.

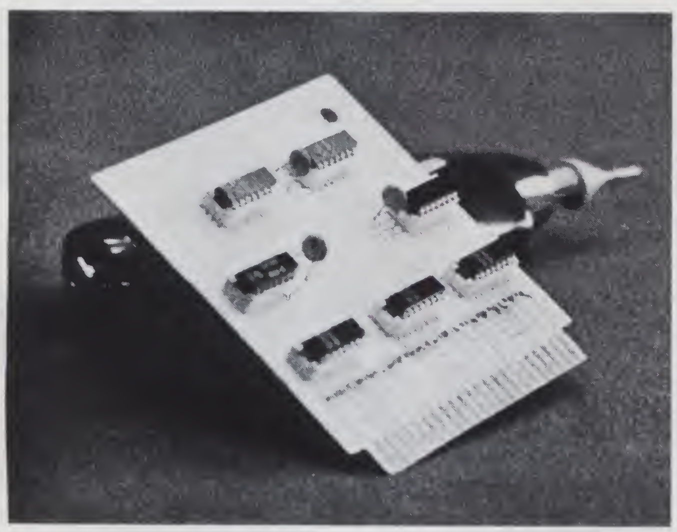


Photo 1. The bus control board is wire-wrapped on an inexpensive plug-in circuit board from Radio Shack.

The address bus is used by KIM to output the address of a memory location with which it wishes to communicate. The data to and from that location is transferred over the data bus. The timing and conditions of these transfers are maintained by the control bus.

The timing is controlled primarily by the phase 1 (Ø1) and Ø2 clocks. The Ø1 and Ø2 signals are essentially square waves 180 degrees out of phase but, in addition, have a less-than-50-percent duty cycle so they are nonoverlapping.

First, let's look at a read operation. When the Ø1 clock goes high, the processor prepares to output the address, which stabilizes on the address bus within 200 ns. Another 100 ns or so is needed for the decoders to select a particular memory chip, and perhaps another 500 ns passes before the memory chip puts out stabilized data. Since nearly 1000 ns exists between the leading edge of Ø1 and the trailing edge of Ø2, it can be assumed that the data will be stabilized before the trailing edge of Ø2. The processor therefore can use this edge to gate the data into its buffer register.

The same procedure can be used during a memory write, except that the trailing edge of Ø2 is used to gate the data supplied to the memory chips by the processor.

The R/W line is an output signal from the processor to tell the memory whether the present operation is a write or a read. A zero on the line indicates a write is to take place. If the inverse of the R/W line and the Ø2 clock are ANDed together, a negative edge is produced coincident with the trailing edge of Ø2, but only if the W/R line indicates a write operation. This signal, called a RAM R/W, is inverted on KIM and provides a positive edge, which latches most standard memory chips.

Most memory chips use either open collector or Tristate outputs (1, 0, open circuit) so that they may have their outputs connected in parallel. This way, chips that are not selected (chip enable pin high) will not produce either a 1 or a 0 out.

Most memory boards will buffer the paralleled chip outputs with drivers. Typically, these drivers will also be Tristate, and only the board selected will output its signals to the bus.

Let's look at a schematic of an S. D. Sales 4K memory board (Fig. 1) to see how it functions.

Address lines 12 through 15 are used to select the board. Since there are 16 combinations of these four lines, 16 4K boards could be used (total 64K of memory). The output of this board decoder (gate 34, pin 6) is used to allow the memory data to enter the bus if the SMEMR (which indicates a read operation on the Altair bus) is high and also (gate 35, pin 12) enables the outputs of a two-input decoder that will

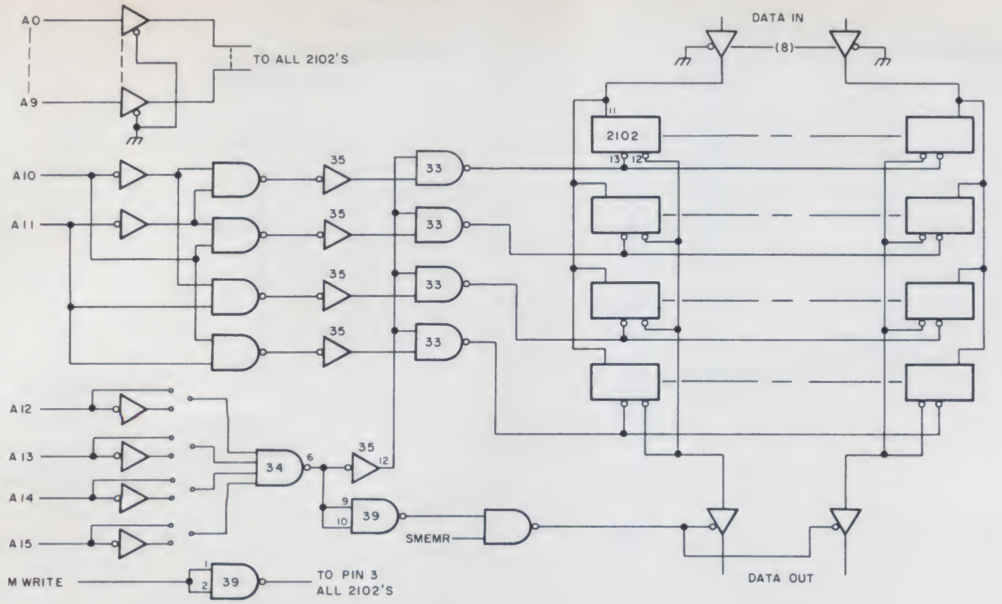


Fig. 1. Simplified schematic of S.D. Sales 4K memory board.

select one of the four banks of memory chips by bringing their chip enables (pin 13) low.

Once a bank of chips has been enabled, the address on lines 0-9 selects one of the 1024 words through internal decoding. Each chip that is enabled "reads" until its pin 3 is brought low, which will cause any data on the inputs to be gated into the locations specified by the address pins. The Altair bus provides this signal, and calls it an MWRITE.

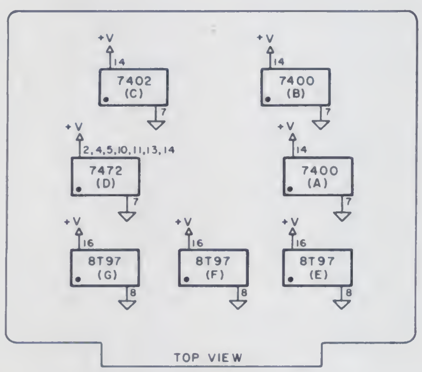
This, or any Altair-compatible memory, can easily be connected to the KIM bus by using the inverse of KIM's RAM R/W as the MWRITE and KIM's R/W for a SMEMR. The address lines are connected KIM AB0 to Altair A0, etc. The data bus is slightly different, as KIM uses a bidirectional bus instead of the two single-direction buses found on the Altair. Since the data lines in and out are fully buffered, just tie the KIM DB0 to both the DO0 and DI0, etc., and memory interfacing is complete.

One of the major differences between the 6502 and the 8080 is the way they handle input/output. The 6502 uses memory-mapped I/O, which simply means that you build a memory location out of flip-flops and output to it with normal memory instructions.

The 8080 instruction set provides separate instructions to manipulate several control lines that in essence are duplicates of the R/W and RAM R/W functions. They are PWR, which acts as a RAM W/R for I/O; SINP, which indicates an input operation; SOUT, which indicates an output operation; and PDBIN, which indicates that the data bus is to be used for an input operation. If it appears that the Altair I/O signals are needlessly complicated, I have to agree; after seeing how the 6502 can simulate the transfers, I'm sure several improvements might come to mind.

When the 8080 performs an I/O instruction, it uses the eight least significant bits as the address of the I/O port. Because of this, the only major change required to convert the 6502 bus is to reserve one page of memory as I/O ports. A decoder must indicate when this page is addressed and output a signal to simulate SINP and SOUT. Page FF (hex) cannot be used since KIM stores its interrupt vectors there; so the next logical choice is page FE.

As long as we've mentioned



AB0	out	A	1	AB0 in
AB1	out	B	2	AB1 in
AB2	out	C	3	AB2 in
AB3	out	D	4	AB3 in
AB4	out	E	5	AB4 in
AB5	out	F	6	AB5 in
AB6	out	H	7	AB6 in
AB7	out	J	8	AB7 in
AB8	out	K	9	AB8 in
AB9	out	L	10	AB9 in
AB10	out	M	11	AB10 in
AB11	out	N	12	AB11 in
AB12	out	P	13	AB12 in
AB13	out	R	14	AB13 in
AB14	out	S	15	AB14 in
AB15	out	T	16	AB15 in
I/O	Enable	U	17	Lower 8K enable
Int.	Enable	V	18	Ram R/W
R/W		W	19	Ready
Sync		X	20	Ø2
Hold	Req.	Y	21	+5 volts
Hold	Ackn.	Z	22	Ground

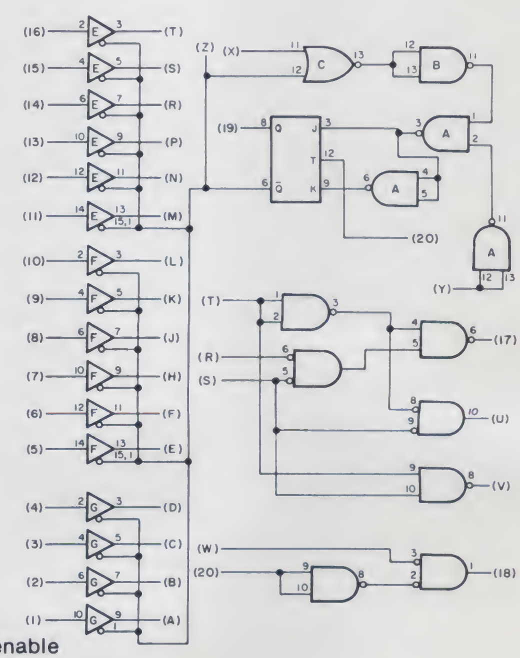


Fig. 2. Schematic, pinouts, and component placement for bus control board.

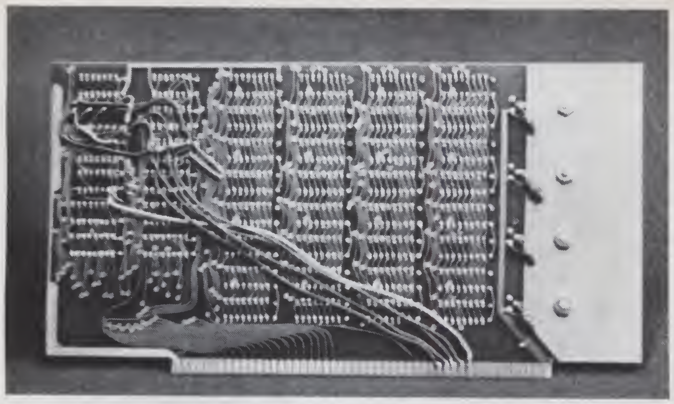


Photo 2. A 4K memory board modified for operation in K1 through K4 using the KIM decoders.

the interrupt vectors, page FF must also be decoded and ORed with the K7 to force the KIM ROM operating system to be executed whenever the RES button on the KIM keyboard is pressed (the KIM monitor vectors are in K7).

To decode both page FF and FE would require two eight-input gates and the appropriate inverters, so I decided to simplify matters. Rather than decode out all eight of the page bits, I checked only AB14 and AB15. If both are high, I assume page FF is being addressed, and if AB14 is low with AB15 high, an I/O operation is indicated. This of course wastes a lot of memory locations. An I/O port, for example, can be indicated by an instruction referencing page 80, 81, 90, AF, as well as 60 others. Actually, the top half of the memory locations cannot be used in this scheme. If you plan to have more than 32K of memory, you should decode out FF and FE with eight-input gates. With the decoding I suggest, you can add a 16K board to your system without additional modifications.

For details of how all the signals previously discussed are implemented, refer to Fig. 2, a reproduction of one page in my notebook. All pertinent information on the bus control board can be read from that sheet, including the schematic, pinouts and component placement. My circuits are all wire-wrapped on 44-pin boards available from Radio Shack. Photo 1 shows the completed bus control board.

I'm usually asked why I did not simply invert the RAM R/W signal from KIM to get the needed RAM R/W. A little inspection will reveal that all 44 pins are being used, and I already had the R/W and Ø2 signals available for the DMA circuits, which are next on the list of topics to be covered.

DMA stands for direct memory access. In general, this means that the processor is forced into a wait, or hold, state while another processor (usually an intelligent controller) manipulates the bus lines to read and write directly from and to memory.

The processor must be effectively removed from both the address and the data bus to avoid conflict with the controller signals. When the 6502 is in a hold state, the data bus goes to an open-circuit condition since it is Tri-stated internally. The address bus, however, must have Tri-state drivers added to it so that when the processor goes into a hold, the drivers can be disabled, leaving the bus free for DMA.

The primary reason for adding DMA capability to the KIM is the Dazzler, which has the capability of mapping memory blocks onto a television screen. When the Dazzler needs to access memory to read a block of data, it issues a request signal and begins the read sequence when it receives an acknowledgment.

A circuit is needed that, upon request from the Dazzler, will put the processor into a hold, disable the bus and generate an acknowledge.

There are several problems, however, that make this circuit more complicated than you might expect. First, the processor will react unpredictably if you try to put it into a hold during a Ø2 cycle. In order to insure that a request will not be recognized during the Ø2 cycle itself, I used a JK flip-flop that changes only on the trailing edge of the Ø2 clock. When this flip-flop clears, the address bus drivers are disabled, the hold line on the processor is pulled low, and an acknowledge is generated.

The second catch is that the processor will not stop if it is performing a write operation. Since the bus has been disabled, the Dazzler could take over for itself without problems. If, however, the processor was attempting to modify data in memory on the Dazzler side of the drivers, that data would not be modified and errors could result.

Therefore, I had to insure that the request could not be recognized unless the processor was in a non-write cycle.

You would expect that the R/W line could be used to determine if the presesnt cycle was read or write. Unfortunately, the R/W line is not valid until well into the cycle. I chose to use the sync signal, which signifies that the processor is fetching an op code. A normal instruction always *reads* an address following an op-code fetch. The only exceptions are single-byte instructions, which *read* another op code in the next cycle. The point is that if the sync line is high, the next cycle must be non-write. I use this principle to ignore the Dazzler's request until the sync is high, thus insuring that the processor will halt when the flip-flop is cleared.

I planned for the first two situations, as they are fairly obvious from a study of the 6502 hardware manual. The third problem, however, cost me a lot of sleep; it was not until I drew a timing diagram extending into the next cycle that I found a solution.

When the processor goes into a hold, the state of the sync line is not guaranteed. It may be either high or low, depending on the type of cycle it is in. If it happens to be low when the processor is in a hold, the next Ø2 pulse would enable the bus again by setting the flip-flop. To prevent this, I ORed the sync signal with Q output of the flip-flop. Even wired this way, the flip-flop will not clear until the sync is high; once cleared, however, it will remain so until the Dazzler ends its request. Without this latching circuit, the video tear is unbearable.

Since my only DMA operation is for the Dazzler, which never writes to memory, my circuits only allow for read-oriented DMA. Write-oriented DMA, such as a floppy disk, would need the R/W and RAM R/W lines Tri-stated along with the address bus. Two Tri-state drivers are available on the bus control board for this opera-

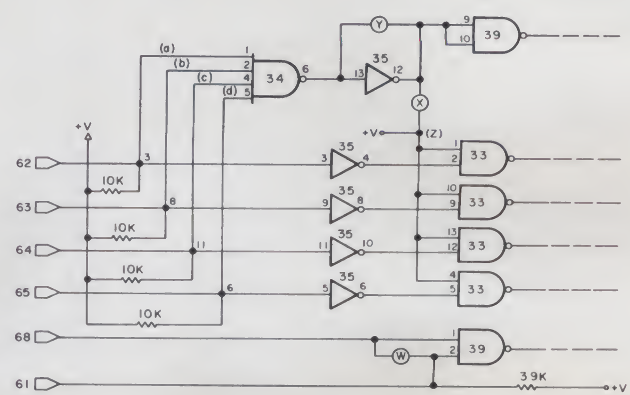


Fig. 3. Modifications to 4K memory board to allow KIM decoding.

tion, but since all 44 output pins are being used, other provisions would have to be made if you decide to add a disk in the future.

Recall that I used the S.D. Sales memory board as an example for explaining the bus conversion. I want to discuss the board again to show how it can be modified to fit in K1 through K4 using the on-board KIM decoder, for those who want this option. Since I designed the system, most of the K1 software has become available in K8 versions, making the following conversion totally a matter of choice. If you do not require memory in the K1 through K4 positions, the S.D. Sales board can be used on the KIM System Altair bus without modification.

Fig. 3 and Photo 2 show the modifications. The foil must be cut at points X and Y. Point Z can be wired to +V by connecting pin 13 and 14 on IC33. A wire must be added to connect pin 12 of IC35 to pin 9 on IC39.

Address	Instruction	Comments
0000	A9 10	Loads Acc. with mode (see Dazzler manual).
0002	8D OF 80	Sends mode to Dazzler input port OF.
0005	A9 90	Loads Acc. with starting address (page 20).
0007	8D OE 80	Sends starting address to Dazzler.
000A	4C 22 1C	Jump to KIM monitor.

Fig. 4. Program to start Dazzler.

ICs 36 and 37 (including the sockets) are not used. Refer to Fig. 3 and connect the points a, b, c and d to the designated pins where IC36 would have been installed, and on the gold fingers for the Altair pins 62 through 65. These were previously connected to the KIM decoder. If you have trouble understanding how the memory works with these modifications, re-read the explanations at the beginning of this article.

A write-protect modification is also shown in Fig. 3. A zero on pin 2 of gate 39 holds its output high. As indicated last month, switch number two under AUXILIARY is used for this feature. The foil cut at point W is on the component side of the board, and is under the socket, so make it before beginning construction.

With the bus control board in place, the KIM should be operational. If you install the memory boards (the 8K board should start at page 20_{16}) and the Dazzler on the Altair bus, the program in Fig. 4 should put a display on the screen. Use the KIM keyboard to write data into page 20 locations, and watch the display change.

Remember, since the 4K board no longer uses the standard address lines, it cannot be used for Dazzler displays.

I should add one comment on the Dazzler. Since my last article, it has come to my attention that random noise can build up on the Altair pins 54 and 99, causing some Dazzlers to turn off suddenly. If you are having this problem, it can be corrected by connecting each of these pins to +5 volts through 2200 Ohm resistors.

Next, I'll discuss the serial/parallel board, which will turn the SWTP printer and keyboard into a miniature teletypewriter. We'll also add an external display and keyboard to KIM, and the front panel will start to come alive.■

Interfacing Tips

communicating with the outside world

Dr. Mark Boyd
St. Mary of the Plains College
Dodge City KS 67801

Microcomputers use TTL-compatible I/O chips. Since TTL logic levels are not directly useful for communication with or control of external devices, special interfaces are required. These interfaces convert TTL logic levels to the signals required by the external devices or convert signals from the external devices to TTL logic levels.

The perfect interface for microcomputer use would interface TTL to anything electrical. Although it is not practical to build such a general interface, it is possible to use the same basic interface design for a wide range of applications. In this article, we will look at basic TTL-to-current and TTL-from-current interfaces. We will then discuss specific applications of these interfaces. These applications include RS-232, TTY, sensing ac or dc voltages and switching ac or dc current.

Two basic components are required in all the designs we will discuss: A TTL Schmitt trigger and an optical coupler. Other components are required to tailor the interface for a particular application, but these two are fundamental to the design. Let's review what they do.

The Schmitt Trigger

The TTL Schmitt trigger can be pictured as a snap-action electronic switch controlled by the current drawn from its input. When no current is drawn, its output is TTL low (0 to .8 volts). When the current exceeds approximately .6 mA, the output switches rapidly to TTL high (2.4 to 5 volts). When the current drawn is reduced below approximately .4 mA, the output switches rapidly back to TTL low. The difference between the two switching thresholds is called the hysteresis range. Since the voltage at the input is about 1.7 volts with .4 mA drawn and about .8 volts with .6 mA, the voltage hysteresis range is approximately .9 volts. The current hysteresis range is approximately .2 mA.

The snap-action and hysteresis characteristics of the Schmitt trigger make it useful for interface design. A normal TTL gate, or inverter, will go into oscillation if its input remains very long in the forbidden .8 to 2.0 volt range. A Schmitt trigger will switch cleanly whenever either of its thresholds is crossed, regardless of how slowly its input is changing. Once the switching has occurred, a large change in the input is required before it will switch again.

TTL Schmitt triggers are available in both regular and LS versions. The LS versions require about 1/3 the input current to switch (.2 mA and .14 mA) but have about the same threshold voltages as the regular version. All the interface designs discussed here use the 7414, a hex, Schmitt trigger input inverter in a 14-pin dip. TTL NAND gates and one-shots are also available with Schmitt trigger inputs.

The Optical Coupler

Optical couplers are available in a wide range of designs. The basic idea is to use an input device to generate light and a light-sensitive output device, which controls an output current in response to the light. Since there is no electrical connection between the input and output, we don't have to worry about differences in voltage or ground conventions. This simplifies interface design.

I have used the MCT-2 optical coupler in all designs in this article. It uses an LED as its input device and a photo transistor as its output device. Other optical couplers with greater speed or sensitivity are available, but the MCT-2 is a good general-purpose device. Higher-speed devices, such as the MCD-2, which uses a photo diode, are needed if you want to use bit rates above 100K bps.

The basic interface circuits for TTL to or from current are shown in Fig. 1. The input circuit uses the photo transistor as a current sink for a Schmitt trigger input. Using a 7414 and an MCT-2, the TTL output will switch high when the current through the LED exceeds approximately 2 mA. The output circuit uses a TTL inverter to sink current from the +5 volt supply through the LED. If you are building both input and output circuits, you will probably have 7414 inverters available for this purpose. The photo transistor can control currents up to 50 mA and withstand more than 30 volts in the off state.

In most applications, current-limiting resistors will be used to limit the LED or photo transistor current to the desired value. Amplification is necessary if the input signal is very small or the output is required to handle large amounts of power. I'll cover ac and dc power control, but not amplification of small signals. I suggest Walt Jung's *IC Op Amp Cookbook* if you need this information.

Let's consider RS-232 and TTY 20 mA current loop interfaces as our first applications. Both of these signals are large enough to drive the input without amplification,

but small enough to be controlled by the photo transistor; therefore, only a few passive components must be added to the basic interfaces.

RS-232

RS-232C standards specify a voltage level of +5 to +15 volts into a 3k to 7k load for logic 0 and -5 to -15 volts for logic 1. Most RS-232 drivers can source far more current than we need, so we must pick a current-limiting resistor to make our input circuit switch with +5 volts in. Since we need about 2 mA, we will pick our resistor to allow 2 mA to flow from a 5 volt source. The voltage drop across the LED is close to 2 volts, so our resistor must have a 3 volt drop across it, with 2 mA flowing through it. Ohm's law gives (3/.002) Ohms or 1.5k Ohms. We must also protect the LED from reverse voltage with a diode across it. Our RS-232-to-TTL circuits (input) is given in Fig. 2a.

For RS-232 output, we need to supply ±5 volts or more into a 3k Ohm load. The output circuit shown in Fig. 2b will supply +7.5 volts for logic 0 and -8.25 volts for logic 1 into a 3k Ohm load. This circuit uses a ±12 volt supply since most microcomputers have these voltages. I suggest you check my figures using Ohm's law; remember, the transistor has about a one volt drop across it when it is switched on.

Current Loop

The TTY input and output circuits, given in Figs. 3a and Fig. 3b, respectively, are my variation on Motorola's circuits as used in the SWTP 6800. I don't have a TTY, so I haven't used these circuits. The input circuit uses a 1k Ohm 1 Watt resistor to limit the current flow (from a 24 volt source) to approximately 20 mA. The capacitor and the 1k Ohm ¼ Watt resistor improve noise immunity. The output circuit also uses a 1k Ohm 1 Watt resistor for current limiting. The diodes protect the LED from reverse voltage and voltage spikes. Both circuits use ±12 volts, with -12 volts as TTY ground.

DC/AC Detection

Now let's look at a more general input circuit. Fig. 4a shows a circuit that will indicate the presence or absence of a dc voltage. By properly choosing the value of the current-limiting resistor, you can use this circuit to detect a few volts to many hundreds of volts. If an input rectifier and filter are added (see Fig. 4b) this circuit can also be used to sense ac voltages. Remember that rectification and filtering will give a dc voltage equal to the peak ac voltage, or 1.414 times the RMS voltage. This circuit will switch its TTL output high when the current through the LED exceeds 2 or 3 mA.

The current-limiting resistor's value can be calculated using Ohm's law. Since the LED has about 2 volts across it, the remainder of the applied voltage must be across the resistor; therefore, the current through the resistor and the LED will be (V - 2 volts)/R. If we know the TTL output will switch high when 3 mA or more current flows through the LED, we can solve for R = (V-2)/.003. Since the actual current required to switch the circuit will depend to some extent on the particular MCT-2 and 7414 used, a variable resistor should be used if accurate voltage-level switching is necessary.

Fig. 4c shows a circuit that will detect the presence or absence of line voltage. I used 120 volts RMS or 170 volts peak for the line voltage and assumed 3 mA of current when I calculated R. This gives a value of (170-2)/.003 or 56000 Ohms. Since the amount of heat generated in the current-limiting resistor depends on the voltage across it, you should use a ½ Watt resistor if R is greater than 10k Ohms, a 1 Watt resistor if R is greater than 30k Ohms, and a 2 Watt or larger resistor

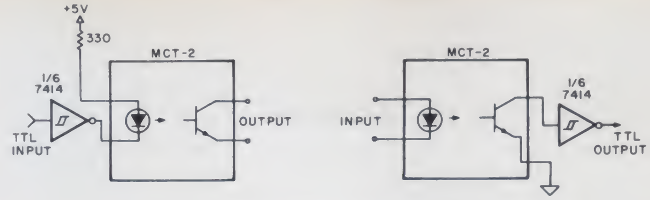


Fig. 1. Basic TTL interface circuits.

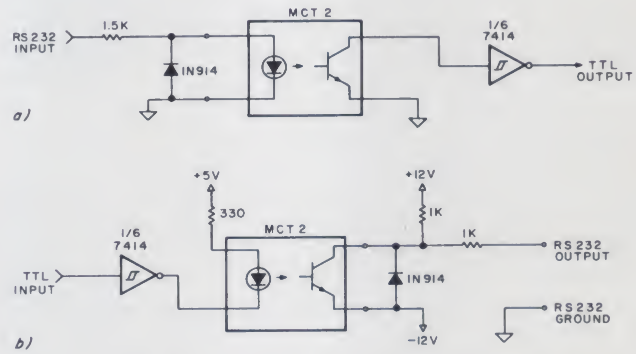


Fig. 2. RS-232-to-TTL input and output circuits.

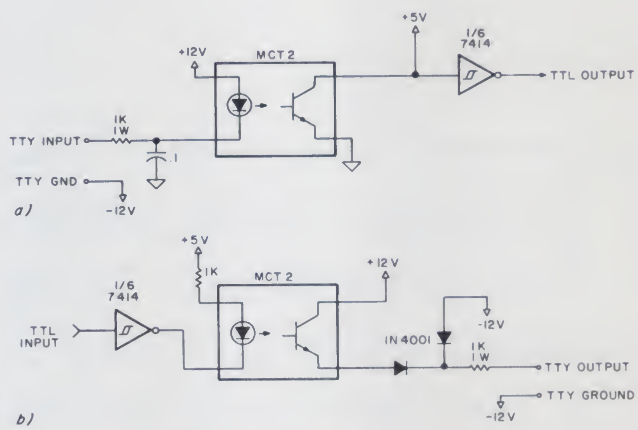


Fig. 3. Teletypewriter I/O circuits.

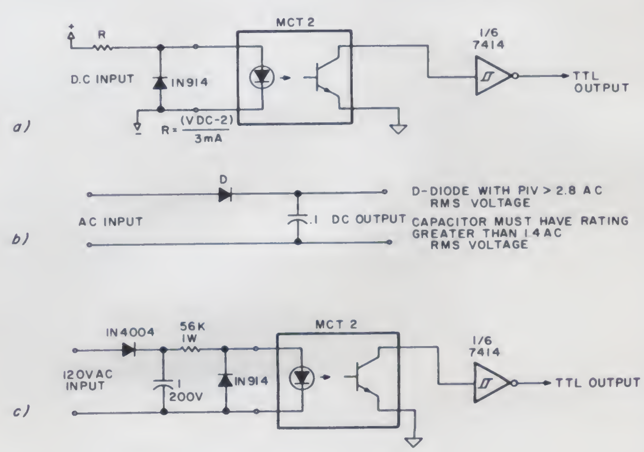


Fig. 4. Dc/ac voltage detection circuits.

if you are monitoring very high voltages. Remember, this circuit draws a couple of milliamps.

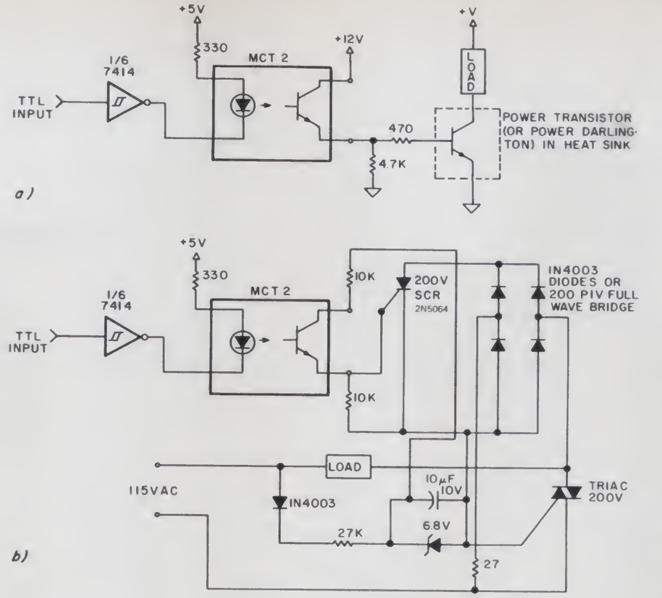


Fig. 5. Power control circuit.

Power Control

Control of dc power is simple. We can use the current controlled by the photo transistor to switch a power transistor or power Darlington between its off state and its saturated state. An example of this approach is given in Fig. 5a. Using an NPN power transistor with a current gain of 30 or more and a control current of 20 mA, this current can control 500 mA or more. By using a power Darlington with a much higher current gain, it can control many Amps. Since the power device is either off or saturated (turned on hard), only minimal heat sinking is required.

An alternate approach is to use the output of the photo transistor to control a relay. This approach works for both ac and dc, but it is not as reliable as a purely solid-state circuit, and it will generate more rf noise. The ac control circuit in Fig. 5b uses the photo transistor to control an SCR connected across a full-wave bridge. Current flow through the bridge and SCR turns on a TRIAC, which allows current to flow through the load. For modest loads the TRIAC isn't needed: just put the SCR and bridge in series with the load. In both the ac and dc power control circuits, a TTL high input to the inverter switches power to the load.

Wrap-up

The circuits in this article are just a few examples of how Schmitt triggers and optical couplers can be used. There are probably better ways to do everything we've covered here, but the simplicity of these circuits makes them ideal for hobbyist use.

Don't limit yourself to using an optical coupler as the input for a Schmitt trigger. Remember it is a current-activated switch. Anything that will sink a few milliamps of current at a volt or two can be used to control it. This includes many kinds of sensors whose resistance depends on light level, temperature, humidity, strain, and so forth. Think of what you can do with all that information! ■

KILOBAUD KLASSROOM NO. 9

Counters and Registers

George Young
Sierra High School
Tollhouse CA 93667

In the last session, we looked at pulses on an oscilloscope and investigated many of the microprocessor support chips that generate and shape pulses. We also looked briefly at timing diagrams, and I gave you a circuit that would generate a hexadecimal number with a switch closure.

Now, we're going to study registers and counters.

Introduction

Counters and registers are built into the microprocessor chip. They also form many of the microcomputer support circuits. We cannot get inside the microprocessor chip to look at these counters and registers, but we can investigate their TTL counterparts to learn how the internal counters and registers must function.

A register is a memory device that is capable of storing a binary digit. Most microcomputers operate on eight bits at a time. Together we call these eight bits a *byte*. In most instances, a microprocessor register will be one or two bytes wide. The term *nibble* has been coined to describe the treatment of four bits together (i.e., a nibble is half a byte).

A counter may be defined as any device that divides an input pulse train (clock) and provides some means to indicate this division.

Experiment #45
The D Flip-Flop

Problem: What is a storage register?

Solution: Let's make one on the console breadboard.

Procedure: Refer to Fig. 1 for the pinouts and test circuits of the 7474 dual D-type flip-flop. Fig. 1a shows the data-manual representation for the 7474. Fig. 1b gives the working drawing of the two flip-flops. Fig. 1c gives two different test circuits for the 7474; either one (or both) of the circuits may be used.

Theory: The D-type flip-flop functions as follows: Whatever voltage level (high or low) is on the D input when you clock, the Clock input will be transferred to the Q output. Since the Q output must be the opposite of the $\overline{Q}$ output, the inverse of the D input will appear at the $\overline{Q}$ output. Slow the console clock down to its slowest speed. Use the console start/stop control to allow you to place either a high or a low on the D input.

The outputs may be monitored with two LEDs as shown in Fig. 1c, or the console logic probe may be used to monitor the Q output, also shown in Fig. 1c. The two-LED monitor version is preferred because we can then use the console logic probe to monitor the clock input signal. This will allow us to observe that the 7474 is a positive edge-triggered flip-flop.

The terminals marked P and C stand for Preset and Clear, respectively. Preset is the same as Set, and Clear is the same thing as Reset. When a flip-flop is set, the Q output is set high. When a flip-flop is cleared, or reset, the Q output is cleared

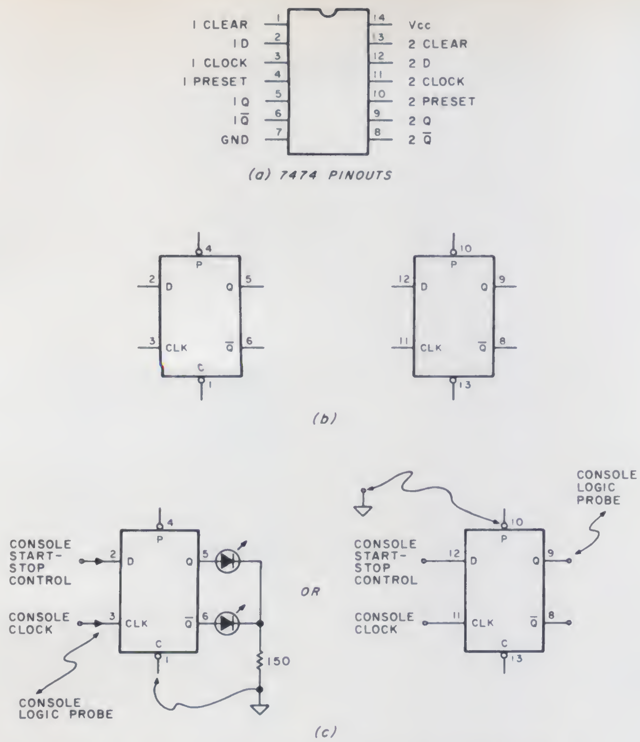


Fig. 1. Pinouts and test circuits for the 7474.

low. Jumper wires are used on the P and C inputs to set and clear the flip-flop. These are active low inputs, meaning that you take the P or C inputs low to accomplish either Preset or Clear. Test your 7474 to make certain that the high or low on the D input is transferred to the Q output as the clock rises, and that the Preset and Clear functions work.

Some circuits may have the $\overline{Q}$ output connected back to the D input of the same flip-flop section. That is, pin 6 is connected to pin 2. Connected in this fashion, the 7474 should toggle (i.e., change state on each clock pulse). Try this on your 7474 and see if it toggles—some do and some don't. Never depend on this test to evaluate a 7474; you might throw away a lot of perfectly good ones.

Experiment #46
7474 Shift Register Configurations

Problem: This circuit shows a 7474 connected in a weird fashion.

Solution: The 7474 may be connected as a shift register. Let's connect the 7474 in a shift-register configuration and see what it does.

Procedure: Refer to Fig. 2. Two 7474s are shown, but the experiment may be done using only one. Connect the input clock pins in parallel. The console start/stop control goes to the first D input on the left. The console logic probe is used to monitor the last Q output.

First, set the console start/stop control low. The low on the D input will transfer to the Q output on the first rising clock edge. The second D input, which is low, will be transferred to the second Q output. The low on the first D input will therefore "travel" down the row of D flip-flops and finally appear at the last Q output on the right.

Now, momentarily ground the Preset input on the first D flip-flop section. This will take the Q output high. With each succeeding clock pulse, this high will be transferred to the next D flip-flop. The single positive pulse will travel from left to right down the chain, changing its location with each rising clock edge. This is depicted graphically in the timing diagram in Fig. 2c.

But, suppose we don't want a positive-going pulse to propagate down the flip-flop chain. Suppose we want a negative-going pulse instead. Fig. 2b shows this connection. The D input is taken high with the start/stop control. The high will then propagate down the chain. We can introduce a negative-going pulse by momentarily grounding the Preset input again, which will set the Q output high and the $\overline{Q}$ output low.

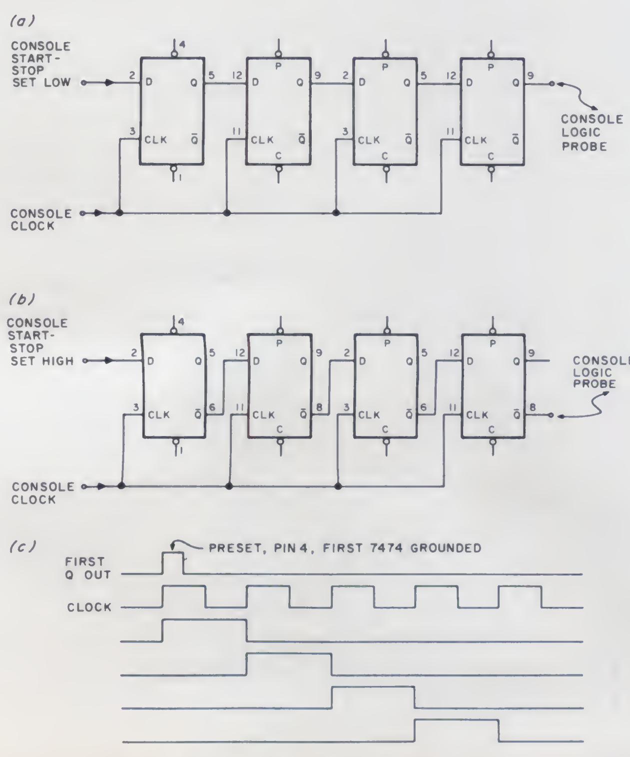


Fig. 2. 7474 shift-register configuration.

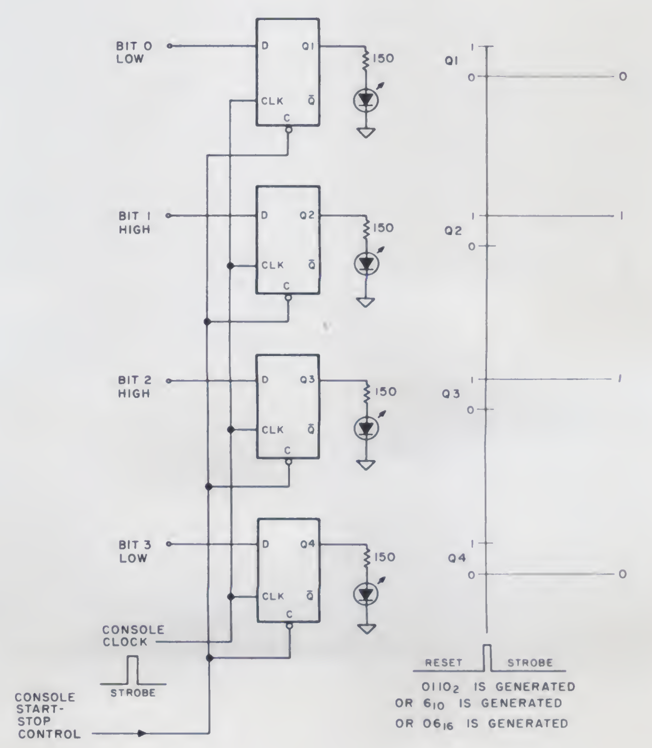


Fig. 3. 7474 data-storage configuration.

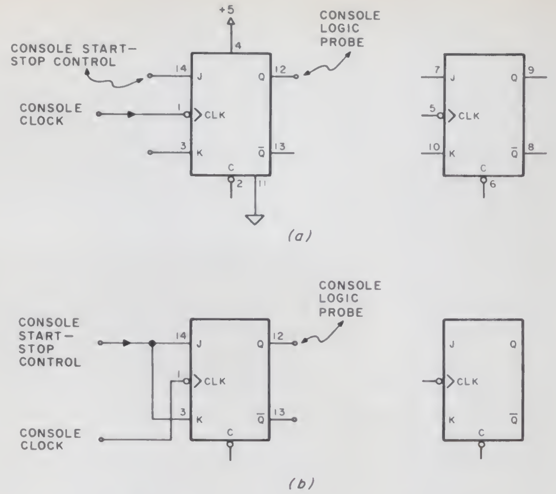


Fig. 4. J K inputs of the 7473.

The low will then propagate down the chain of flip-flops.

Experiment #47 The D Flip-Flop Storage Register

Problem: How can the D flip-flop be used to store binary data?

Solution: Connect the D flip-flop on the console in the storage-register configuration.

Procedure: Fig. 3 shows two 7474s connected in a storage-register configuration. If you only have one 7474, then only half of the circuit is to be used for this experiment. All the clock inputs should be connected in parallel. (Note: they should still be properly connected from the preceding experiment.) Four LEDs are used now to monitor the Q outputs. A 150 Ohm resistor is shown connected in series with each LED. This resistor is not critical. Its purpose is to limit the current through the LED. It may have any value between zero Ohms and about 1000 Ohms. (That's right—you can even leave it out if you want to.) In a permanent circuit, this resistor is necessary, but for our experiments, it may be omitted.

The console start/stop circuit is used here to provide a reset function. The console clock is used to clock the data on the D inputs into the register. The clock signal will often be called a *strobe*, so we have marked it on our experimental diagram as such.

The data inputs are now called bit 0, bit 1, bit 2, and bit 3, and we are starting to use terminology that will be used in the microcomputer. Again, whatever is on the D inputs will be transferred to the Q outputs when you strobe the flip-flops.

We can encode the data inputs as follows: Ground the bit 0 and bit 3 inputs with a jumper wire from these two pins to minus on the breadboard. The bit 1 and bit 2 D inputs are normally floating at a TTL high, so nothing need by done to them for this experiment. Reset the Q outputs low with the start/stop control. On the next rising clock edge, the LEDs will display 0110, which is a binary 6 in both decimal and hexadecimal. By changing the inputs to the D inputs, any decimal number from 0 to 15 may be generated, and the LEDs will display the binary equivalent. The group of four D flip-flops will *store* the binary number until the input data is changed, or until the register is cleared with a reset pulse.

Experiment #48 The JK Flip-Flop

Problem: On this circuit diagram, the J and K inputs of the 7473 are connected.

Solution: Let's get the 7473 back on the breadboard and investigate those J and K inputs.

Procedure: We said in Klassroom No. 3 that we would eventually get to the J and K inputs on the 7473. Let's get two 7473s out of storage and back onto the breadboard. A reminder: Power is pin 4 and ground is pin 11 on the 7473—not 14 and 7! Refer to Fig. 4a. The console clock goes to pin 1, the start/stop control to pin 14, and the console logic probe to the Q output, pin 12.

Theory: Whatever is on the J input is transferred to the Q output as the 7473 is clocked, and whatever is on the K input is transferred to the $\overline{Q}$ output. Sound familiar? With the connection in Fig. 4a on the breadboard circuit, the Q output should follow whatever you put in on the J input of the 7473 as the clock strobes in the data. Move the start/stop control from the J input to the K input. Now, whatever you put in on the K input will appear at the $\overline{Q}$ output of the 7473.

Now connect the J and K inputs together as shown in Fig. 4b and take the console start/stop control *low*. Oops! What's this? We bombed out! It doesn't work like a D input. Now take the start/stop control *high*. Hey! Now it is operating like an ordinary flip-flop. If the J and K inputs are left floating, they assume a floating TTL high. If both the J and K inputs are taken low simultaneously, you "kill" the flip-flop. So, if you want to use the J and K inputs on the 7473, use one J or one K input, but don't ground both unless you want to disable the flip-flop.

Experiment #49 The 74175 Quad D Flip-Flop

Question: Do IC manufacturers make other chips with more functions in them?

Answer: Yes. Let's see what's available.

Procedure: Two 7474s can be replaced by one 7475 or 74175. We will use the 74175 here, although the 7475 may be used if you have that chip available. Fig. 5 gives the pinouts for the chip.

Connect the 74175 on the breadboard in the same configuration we used for the 7474 in Experiment #47 (see Fig. 6). Hang LEDs on the Q outputs and repeat the procedures of Experiment #47.

Theory: The number of pins available on an IC package will, to some extent, determine what can be implemented on a single chip. If the single 74175 is compared with two 7474s, we can give a little more meaning to the preceding sentence. By using two 7474s, we can have four D flip-flops; but we also have the capability of presetting or clearing each flip-flop individually. However, if we use a 74175 we give up this capability and can only clear *all* the flip-flops. If we don't need a particular set or clear feature we can save some PC board real estate and use one chip instead of two.

I need to point out something else to you: Look at Fig. 5. You will note two inverters in the pinout diagram. One inverter is on the Clear line, and the other is on the Clock line. The purpose of these two inverters is to present one TTL load to the driving circuit.

Each inverter drives either four clock circuits or four reset functions inside the chip. If the two inverters were not included in the package the driving circuit would have to drive four TTL loads each instead of only one. We have not had to pay any attention to *loading* con-

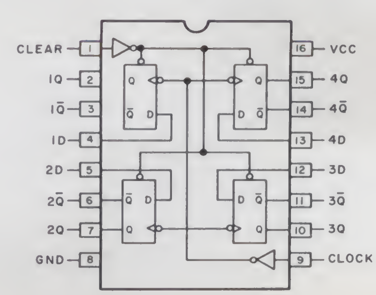


Fig. 5. 74175 pinouts.

siderations so far in this series. Very shortly, we will find that we must pay close attention to this factor in our computer. The "on-chip" inverters are called buffers and are there to reduce the loading on the circuits that drive these inputs.

Experiment #50
The 7473 Binary Ripple Counter

Problem: I need to divide a train of pulses by 8 (or 16) but I don't have an IC that will do the job. Should I buy one?

Solution: No. You already have the necessary ICs in your parts box.

Procedure: Connect two 7473s as shown in Fig. 7. You already set up this circuit back in Klassroom No. 3, but it won't hurt to do it again. I have omitted the pin connections purposely so you can practice looking them up, getting them into the circuit, and working with the chips.

A flip-flop produces one output pulse for each two input pulses. It divides the input clock by two. Since we can monitor the Q outputs with LEDs, we have a device that will count two input pulses and display one output pulse—we now have a counter. Two flip-flops will count to 4, three to 8, and four flip-flops will count to 16. Hang LEDs on each Q output.

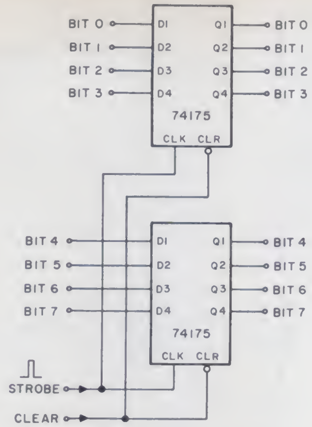


Fig. 6. 74175 data-storage configuration.

If the LED connected to the first output, called A, is positioned nearest you, and the LED connected to D is positioned farthest from you, the display will count in binary format; the least significant bit will be closest to you and the most significant bit farthest away. The display readout is now correct for a vertically read format. If the entire console is rotated 90 degrees counterclockwise, the display will be correct for a horizontal format and is identical to the LED displays on the Imsai and Altair front panels.

The 7473 is called a *ripple*, or asynchronous, counter because the first flip-flop is clocked and each succesive counting stage is clocked from the preceding Q output. The count *ripples* down the counting chain. We'll call the outputs of the counters A, B, C and D. A will be the first Q output, B the second, C the third and D the fourth Q output.

Summary

We have discussed many of the TTL counters and designed experiments around them. I have attempted to give you some idea of shift registers and their operation. You've been building your skill in reading circuit diagrams and getting familiar with some of the discrete chips used for support of the microprocessor chip in microcomputers.

If you compare where you are now with where you were last spring, you should be proud of what you have accomplished. However, judging from some of the reader feedback, I must conclude that some of you have not been doing a very good job with your homework. But, this has always been so in education! Ah, well, nobody claimed that this was going to be easy.

Preview

In Klassroom No. 10, we will complete our general discussion of basic hardware chips with some work on decoders. You should already have some gates and inverters, and you will need an open collector NAND gate (7401 or 7403). You'll also need a 1:8 decoder such as the 7442, 7445 or 74145, or the 8000 series 1:8 decoder, the 8250. For a 1:10 decoder, the 7441, 7442, 7445, 74145 or 8251 will suffice. The 74154 1:16 decoder will also go on the breadboard. Additionally, we will experiment with a Tri-state, or three-state, device such as 8T09 or 8T97 or equivalent.

Sierra Electronics, Box 11, Auberry CA 93602, will make up a suitable experimental package for $5 postpaid in the U.S. and Canada. California residents must add 6 percent sales tax. ■

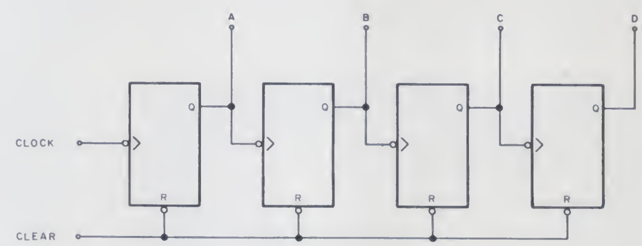


Fig. 7. 7473 binary ripple counter.

Kid Korner

Teaching Preschoolers Letter Discrimination

John Eric Victor
John Victor Associates
11 Idar Court
Greenwich CT 06830

Why not have computers as a family hobby? This question was posed in *Kilobaud* recently (Letters, No. 12) by a mother of two children, ages five and six. The family purchased an IBM 5100 microcomputer, ostensibly for business applications; but in a scene that must have been repeated a thousand times in computer-hobbist homes across the country, the kids just had to get at the computer terminal and make it do something. What's fantastic is that most children cannot help but benefit from their computer experiences. The home computer has to be the greatest "teaching machine" ever invented.

Of course, parents could just sit each child down at the terminal and let him or her plunk away, and possibly the child might come up with something. However, the child will get much more out of the computer experience if the parent can guide the activities.

The following is the first in a series of articles designed to help parents get their children involved in home computing. If *Kilobaud* readers have any ideas, projects, experiences, etc., involving their own children and home computers, I would appreciate hearing from them. The suggestions need not be elaborate.

Ryan Rembaum, age 3, learning to discriminate letters using the author's program. Cardboard template makes it easier for the child to find the letters.

A Letter-Discrimination Program for 3-5 Year-olds

Before children can learn to read—in fact, before they can even name the letters—they must learn to "see" the shapes of letters. Children who are not trained to discriminate letter shapes often ignore details that make one letter different from another. They also tend to view letters turned around or upside down. Telling a p from a d may sound trivial to adults, but most children have considerable difficulty with that type of distinction. Before children are taught to name the letters of the alphabet, they should be given practice matching letter shapes.

Children do learn some letter discrimination from educational television programs such as "Sesame Street."

However, television, even educational television, is a passive medium. The child watching is simply a spectator. A computer program makes the child a participant in the educational process, and the evidence is that people are able to learn much more by being participants than by being spectators.

The program in this article is based on one for early-childhood use that I recently designed for an encyclopedia publisher. That program made use of a "magic pen" that lighted when the child correctly matched letter forms. This *computer* version is written in Apple-I BASIC, but can easily be modified to run with any BASIC interpreter that can handle string input and output to a video terminal. The Apple computer is well suited for this since its video generator produces fairly large letters. A video terminal that can handle both uppercase and lowercase letters is most desirable, if the characters are not too small (as is the case with an 80-character-per-line system).

The idea of the program is to have the child match, on the keyboard, a letter or group of letters that appears on the video screen. If the child is successful, a line of stars shoots across the screen, which is then cleared, and a new letter, or letter group, is presented.

The program works on eight different levels. At the lowest level, the child works with just two letters: A and S. These were picked because they are quite dissimilar, and they happen to be right next to each other on the terminal keyboard. As the child gains skill at one level, he or she moves to another level that presents new letters. At level 6, the child is working with all 26 letters.

At level 7, the child has to match the letters in a two-letter word; at level 8, the word size is increased to three. Remember: The child is not learning to read, but is simply matching letters in the prescribed sequence.

The importance of keeping the program simple and free of unnecessary "stimulation" cannot be overemphasized. Most of the pictures and artwork found in children's books are, in fact, designed to appeal to the adults buying the books. Children have trouble working with pictures that contain many elements. This program is designed to show only the letters in question in the middle of a blank screen to eliminate distractions. The input appears at the bottom of the screen to keep it away from the letter being matched in the center.

If the child makes a mistake, or inputs too many letters, not much happens on the screen. The correct input produces all of the action. The idea is to reward success and not failure. Making mistakes too distinctive (a beep, Xs or some other output for a wrong input) tends to emphasize failure and may make the child anxious.

The terminal keyboard may present a problem to a young child who might see it as a bewildering maze of symbols and levers. As an analogy, picture the scene from the old James Bond movie *Goldfinger*, in which Bond has just "eliminated" his arch foe Odd Job and must now deactivate an atomic time bomb set to blow up Fort Knox. He has only minutes. He opens the bomb casing and is confronted by a complex array of wheels and instruments, all whirling away at a furious rate. The nearly omniscient Bond

One of the author's computer systems—a 6502-based Apple-I microcomputer with 8K RAM. The monitor is a converted Hitachi.

```
10 REM LETTER DISCRIMINATION PROGRAM IN
15 REM APPLES BASIC FOR 3 TO 5 YEAR OLDS
20 REM BY JOHN ERIC VICTOR
30 DIM A$ (50), B$(5), C$(5)
35 REM PARENT INPUTS
40 PRINT "TO CHANGE LEVEL INPUT A '+'"
50 PRINT "TO END PROGRAM INPUT A '!'"
60 R=1: PRINT: PRINT "WHAT LEVEL DO YOU WANT? (INPUT NO.)"
70 PRINT "1=A,S": PRINT "2=A,S,D": PRINT "3=A,S,D,Q,W,E,Z,X,C"
80 PRINT "4=E,F,R,T,G,V,B,Y,H": PRINT "5=U,I,O,P,J,K,L,N,M": PRINT "6=ALL LETTERS"
90 PRINT "7=2 LETTER WORDS": PRINT "8=3 LETTER WORDS"
95 REM SELECTION OF LEVEL
100 INPUT A
110 IF A=1 THEN A$="AS": IF A=2 THEN A$="ASD"
120 IF A=3 THEN A$="ASDQWEZXC"
130 IF A=4 THEN A$="EFRTGVBYH"
140 IF A=5 THEN A$="UIOPJKLNM"
150 IF A=6 THEN A$="QWERTYUIOPASDFGHJKLZXCVBNM"
160 IF A=7 THEN A$="ISASNOTOATITBYBEMEWEOHHIHO"
170 IF A=8 THEN A$="FARTARRATROTRUMMENNEWNOWWONAIREARPEATONNOT"
175 REM SELECTION OF LETTERS
180 IF A=6 THEN 400: IF A=7 THEN 410: IF A=8 THEN 420
200 REM PRINT ONE LETTER OR WORD ALONE IN CENTER OF SCREEN
210 GOSUB 500
230 TAB 20:PRINT B$: GOSUB 500
240 REM THE CHILD RESPONDS
250 TAB 19: INPUT C$
260 IF C$="+" THEN 60: IF C$="!" THEN END
270 REM CHILD MAKES ERROR
280 IF C$ # B$ THEN 250
290 REM CORRECT RESPONSE
300 FOR P=1 TO 40: PRINT "*";: NEXT P: GOTO 180
390 REM SELECTION OF LETTER OR WORD FROM 180
400 N=LEN(A$): R=RND(N)+1: B$=A$(R,R): GOTO 210
410 B$=A$(R,R+1): R=R+2: IF B$="HO" THEN R=1: GOTO 210
420 B$=A$(R,R+2):R=R+3: IF B$="NOT" THEN R=1: GOTO 210
490 REM SPACING
500 FOR P=1 TO 12: PRINT: NEXT P: RETURN
```

Program listing.

simply stands in front of the device dumbfounded. This gives you an idea of how a three-year-old feels when first confronted by a computer terminal.

We can get around this problem by covering the keyboard with a piece of cardboard that has a window cut out to show only the keys the child will be using. Also, the adult supervising the child should push the RETURN key for the child.

As the child progresses, more and more of the keyboard will be exposed, until he or she is using the entire keyboard. Later on, the child will be able to use the program without help.

Children have a tremendous tolerance—in fact a definite need—for repetition. It will not hurt to have the child repeatedly work at any easy level. However, pushing a child to a level that is too difficult will quickly kill his or her interest in computers. Some parents tend to overestimate their children's skills and abilities. Admitting that their child is "slow" seems to be a bitter pill for many parents (two geniuses, Thomas Edison and Albert Einstein, were very slow learners as children). It is best to let the child work at his or her own pace, and at a level that does not allow production of too many errors. ■

```
TO CHANGE LEVEL INPUT A '+'
TO END PROGRAM INPUT A '!'
WHAT LEVEL DO YOU WANT? (INPUT NO.)
1=A,S
2=A,S,D
3=A,S,D,Q,W,E,Z,X,C
4=E,F,R,T,G,V,B,Y,H
5=U,I,O,P,J,K,L,N,M
6=ALL LETTERS
7=2 LETTER WORDS
8=3 LETTER WORDS
?7
                    IS


                   ?SI
                   ?IS
      ***************************************


                    AS
```

Fig. 1. Sample run.

Dr. Lance A. Leventhal
PO Box 1258
Rancho Santa Fe CA 92067

Why Structured Programming?

some insights to consider

Most programmers soon find that writing a large program is much harder than writing a series of small programs. A 2K program isn't merely twice as difficult to get working as a 1K program; it's five or ten times as difficult. If you just sit down and start writing a large program, you can easily feel as if you're fighting an octopus. Every time you get one tentacle under control, another one is trying to do you in. You don't have to be a brilliant theoretician to realize that a large program requires careful organization and systematic procedures. The question is how to achieve this organization, i.e., how to plan your approach before tackling the problem.

One method that allows programmers to get a handle on large problems is called structured programming. Large computer installations use this method in software projects such as airline reservations systems, automated warehouses, data base management systems and communications networks. These projects usually involve many programmers and tens of thousands of instructions. We will discuss the aims of structured programming, the techniques involved and some examples of its use in the type of programs you might want to write.

Aims of Structured Programming

Structured programming is a technique widely used by professional programmers. Its aim is to reduce the cost of software development by making program structure simpler and thereby making programs easier to debug. This is vital in large-scale software since the size of the programs makes them impossible to manage otherwise. But easier debugging is also important to hobbyists, even though they seldom write very large programs. The problem for hobbyists is that their primitive debugging tools make finding and correcting errors difficult. Therefore, methods that can significantly reduce the number of programming errors can greatly reduce the time required to make programs run. For example, a chess program could easily occupy 12K of memory. Certainly you would want to plan such a program carefully before settling down to write the instructions.

Methods of Structured Programming

Structured programming is a systematic approach to producing programs with a simple sequence of operations in which errors can be precisely located and eliminated before you actually write any computer instructions (*code* the program). Its principles are (see the references for a more extensive description):

1. Only a limited but complete set of program structures are permitted. A "structure" is a specified sequence of operations like those shown in Figs. 1 and 2.
2. Each structure has a single entrance and a single exit.
3. No unconditional transfers of control (GOTOs or jumps) are permitted.

A structure may refer to a subroutine, but the subroutine must act like a single instruction, i.e., control passes to the subroutine and back to the main program just as if the subroutine were one instruction.

The ideas behind structured programming are really quite simple. Obviously, the logic of a program will be much easier to check and correct if the program proceeds in a straight line rather than jumping around wildly. Furthermore, if you only use a particular set of structures, you will find it easier to ensure that the logic is right. After a while, the structures become familiar, much like standard phrases or musical patterns. But note right away that you must give up some flexibility in order to achieve this familiarity. Some problems will not fit the structures well at all. Then you may feel as if you're trying to thread a needle while wearing boxing gloves. Don't forget that any programming method involves trade-offs.

The most popular set of structures is the following (theorists have proved mathematically that all programs can be written in terms of these three structures, i.e., they are a *complete* set).

1. The linear structure consisting simply of consecutive instructions or structures. The computer executes these in the order in which they are written.
2. The IF-THEN-ELSE conditional structure, i.e., IF A THEN P1 ELSE P2 where A is a condition and P1 and P2 are programs. Fig. 1 shows the logic of this structure.
3. The DO-WHILE loop structure, i.e., DO P WHILE A where A is a condition and P is a program. Fig. 2 shows the logic of this structure.

A program may consist of a single structure or any combination of structures, all of which belong to the limited set.

In the conditional structure, the computer executes P1 if A is true, P2 if A is false. The programmer can omit P2 if it is unnecessary — if the computer is to do

```
if DATA ≥0 then ABSVAL = DATA else
ABSVAL = -DATA
```

Example 1. P2 is included.

```
if CENTS ≥50 then DOLLARS = DOLLARS +1
```

Example 2. P2 is omitted.

```
COUNT = 0
do while COUNT <10
    BUFFER (COUNT) = 0
    COUNT = COUNT + 1
    end
```

Example 3. Clearing 10 memory locations starting with BUFFER (0).

```
DATA = 0
COUNT = 0
do while DATA ≠CARRIAGE RETURN
    read DATA
    LBUF (COUNT) = DATA
    COUNT = COUNT + 1
    end
```

Example 4. Reading ASCII data and storing it in a line buffer (LBUF) until the operator presses the carriage-return key.

nothing if A is false. The statement in Example 1 sets ABSVAL to the absolute value of DATA; it includes two alternative actions.

The statement in Example 2 rounds DOLLARS to the nearest dollar. No action is necessary (or taken) if CENTS is less than 50.

In the loop structure (see Example 3), the computer first evaluates the condition A. If A is false, P is not executed at all. If A is true, the computer executes P and returns to check A again and so on (the computer repeats P until A becomes false). An alternative is the DO-UNTIL structure in which the computer repeats P until A becomes true. END marks the end of the loop structure, i.e., the statements between the DO-WHILE and END form program P.

Referring to Example 4, note that the initial statement, DATA = 0, is necessary so that the program will be executed the first time, even if the undefined variable DATA contains a carriage-return character. The restrictions on the allowed structures make the extra statement necessary. This inefficiency is part of the cost of the increased simplicity and clarity of structured programming.

Of course, the program could be very complex with many *nested* conditional or loop structures — structures that contain other similar structures. For example, the program for a retail sales terminal contains many conditional and loop structures in order to handle a variety of complex transactions. The programmer can make it easier to see where the various structures begin and end as follows:

1. Indent to separate the current structure from the previous conditional or loop structure within which it is nested.
2. Use END to mark the completion of a loop structure.
3. Use ELSE alone, ELSE NULL, ENDIF or FI (IF backwards) to mark the completion of a conditional structure.

We will use indenting of nested structures and an END with each loop structure but will not mark the completion of conditional structures.

You should note the kinds of programs that structured programming does not permit. Fig. 3 shows a program that includes an extra entry point. Now, if you find an error in P_1 you must first determine how the computer got to P_1. Did it come down through the top or did it sneak in from the side? If it entered from the side, where was it before? And if you make one path work, will the other one still work?

Fig. 4 flowcharts a program with an extra exit. If you have to change something here, you will have to make sure that both exits work properly. And you'll also have to mark where each exit is.

Of course, sometimes programs like those in Figs. 3 and 4 will save you some time or memory — but are those savings worth it? Not unless your program is really time-critical or that extra memory is more than your system has. Remember: Testing and debugging are the difficult stages. A slightly faster or shorter program often isn't worth the extra time required. Following the rules of structured programming takes some discipline, but the payoff can be large. Besides, you can always improve the program once it is working if you are one of those people who would rather tinker with the program than use it.

Language

Structured programming does not assume any particular computer language. The programmer should, of course, be consistent but need not define the rules of the language in detail. The aim is to produce a clear statement of the program logic that you can check before you write any actual computer instructions.

If the language that you are using contains the required structures, you may enter the structured program into the computer directly. Some high-level languages, e.g., PL/I or PASCAL, actually have the required structures. Unfortunately, hobbyists seldom have access to such languages. Programmers who use other languages must translate the structures into simpler statements or into assembly or machine-language instructions. One of the great weaknesses of BASIC is that it lacks structures. Many man-

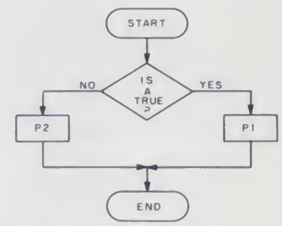


Fig. 1. Flowchart of the IF-THEN-ELSE conditional structure.

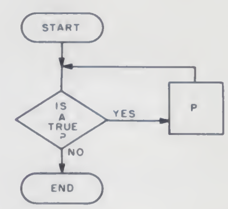


Fig. 2. Flowchart of DO-WHILE loop structure.

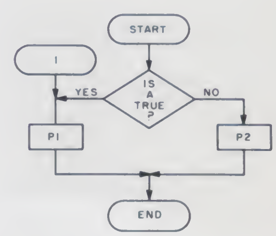


Fig. 3. An illegal program with an extra entry point.

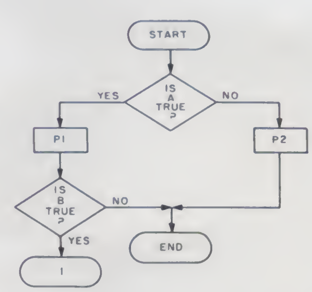


Fig. 4. An illegal program with an extra exit.

```
*
* GENERATE TWO RANDOM ONE-DIGIT NUMBERS
*
N1 = INT (10 * RND (0) )
N2 = INT (10 * RND (0) )
SUM = N1 + N2
*
* STATE QUESTION AND FETCH STUDENT'S ANSWER
*
PRINT 'PROBLEM IS', N1, '+', N2, '='
INPUT ANSWER
*
* CHECK ANSWER AND RESPOND
*
if ANSWER = SUM then PRINT 'CORRECT' else PRINT 'WRONG'
```

Fig. 5. A program that produces a single one-digit addition problem.

```
*
* CLEAR NUMBER OF PROBLEMS, NUMBER CORRECT
*
CORRECT = 0
COUNT = 0
*
* 10 1-DIGIT ADDITION PROBLEMS
*
do while COUNT ≠ 10
    COUNT = COUNT + 1
    N1 = INT (10 * RND (0) )
    N2 = INT (10 * RND (0) )
    SUM = N1 + N2
    PRINT 'PROBLEM IS', N1, '+', N2, '='
    INPUT ANSWER
    if ANSWER = SUM then
        CORRECT = CORRECT +1
        PRINT 'CORRECT'
    else PRINT 'WRONG'
    end

*
* PRINT SCORE
*
PRINT 'SCORE IS', CORRECT, 'OUT OF 10'
```

Fig. 6. A program that produces ten one-digit addition problems.

```
*
* CLEAR NUMBER OF PROBLEMS, CORRECT ANSWERS
* SET CONTINUE FLAG TO GENERATE PROBLEMS
*
CORRECT = 0
COUNT = 0
CONTINUE = 1
*
* GENERATE 1-DIGIT ADDITION PROBLEMS UNTIL CONTINUE FLAG CLEARED
*
do while CONTINUE = 1
    COUNT = COUNT + 1
    N1 = INT (10 * RND (0) )
    N2 = INT (10 * RND (0) )
    SUM = N1 + N2
    PRINT 'PROBLEM IS', N1, '+', N2, '='
    INPUT ANSWER
    if ANSWER = SUM then
        CORRECT = CORRECT + 1
        PRINT 'CORRECT'
    else PRINT 'WRONG'
*
* CHECK IF STUDENT WISHES TO CONTINUE
* IF NOT, CLEAR CONTINUE FLAG
*
    PRINT 'DO YOU WISH TO CONTINUE?'
    INPUT ANSWER
    if ANSWER = 'N' then CONTINUE = 0
    end
*
* PRINT SCORE
*
PRINT 'SCORE IS', CORRECT, 'OUT OF', COUNT

Note: 'N' is the character N.
```

Fig. 7. This program produces one-digit addition problems until the student requests it to stop.

ufacturers are now working on versions of PASCAL, a new general-purpose language specifically designed for structured programming. PASCAL may supplant BASIC as hobbyists move into more complex programs.

Other languages with structures may be available either now or in the near future, particularly if your computer is compatible with one of the widely used mini-computers. For example, the Pacific Cyber/Metrix PCM-12 will run any language that is available for the DEC PDP-8. Similarly, the Heathkit H11 will be able to run many of the languages available for the DEC PDP-11.

Advantages of Structured Programming

The advantages of structured programming are:

1. You can check the logic of the program on test cases by hand since the sequence of operations is simple. The program does not jump around wildly.
2. You can be sure that any program can be formulated in this way.
3. The structured program serves as documentation. Remember, it is language and processor independent. Basically, structured programming forces programmers to define the logic of their programs clearly. The result is fewer errors for you to find in the debugging stage.

Note the importance of the structured restrictions on the sequence of operations. When you are checking the program you never have to worry about some other entry point that could have brought you to the same place in the program. Each structure has only a single entrance and a single exit. So, corrections cannot introduce errors into sequences of operations, and you can make changes far more easily than in an unstructured program.

Structured Design of a Math Quiz Program

Let us now look at the structured design of a program that provides simple mathematics problems to test skills. We will use our own language and will not closely define it other than by saying it contains the required structures. The emphasis will be on clarity and simplicity rather than on program efficiency. We will assume some features of BASIC such as a random number generator (RND), an integer function (INT) and INPUT and PRINT statements. An asterisk (*) will indicate a line of comments.

Provide a single one-digit addition problem.

The program in Fig. 5 contains a single IF-THEN-ELSE conditional structure that selects one of the two responses to be printed. The condition is ANSWER = SUM; the two alternative programs are simply the two PRINT statements. You may want to try implementing the structure in whatever computer language you use. The rest of the program simply consists of statements that the computer will execute

sequentially. Note that there are no unconditional jumps in the structured program. However, you may need some in your assembly language or BASIC program.

Introducing a Loop

Provide ten one-digit addition problems.

The program in Fig. 6 extends the one in Fig. 5. Now the IF-THEN-ELSE conditional structure of Fig. 5 is part of the program within a DO-WHILE loop structure. Note that we must initialize COUNT to 0 before entering the loop. Note also that the first program in the conditional structure consists of two statements, one of which counts the number of correct answers. The indenting clarifies which statements are part of structures. Try to write this program in BASIC or assembly language: Do you think that the structured program is easier to follow and understand?

Adding Another Conditional Structure

Provide one-digit addition problems until the student asks the computer to stop.

The program in Fig. 7 is a little more complex than the one in Fig. 6. Now the computer asks the student after each problem if he or she wishes to continue. The DO-WHILE loop structure forces us to assign a starting value of 1 to CONTINUE so that the loop will always be executed the first time. Here, the program would be more efficient if we had a structure (like the FORTRAN DO statement) that checked the condition after, instead of before, executing the loop.

This program has two conditional structures within the loop structure. The second has no ELSE (why?). What happens if the student accidentally enters some letter besides N or Y? Now would you change the program so that it interprets any response besides a Y as negative? How would you change the program so that it rejects a response other than Y or N and repeats the question? Note that once you get used to the structures, the programs really are easier to understand and correct than are unstructured programs. Often logical errors or omissions become immediately obvious when you write the structured program.

```
*
* CLEAR NUMBER OF PROBLEMS, CORRECT ANSWERS, FIRST-TIME ANSWERS
* SET CONTINUE FLAG TO GENERATE PROBLEMS
*
CORRECT = 0
FIRSTTRY = 0
COUNT — 0
CONTINUE = 1
*
* GENERATE PROBLEMS UNTIL CONTINUE FLAG CLEARED
*
do while CONTINUE = 1
    COUNT = COUNT + 1
    N1 = INT (10 * RND (0) )
    N2 = INT (10 * RND (0) )
    SUM = N1 + N2
    PRINT 'PROBLEM IS', N1, '+', N2, '='
*
* WAIT FOR 3 TRIES OR CORRECT ANSWER
*
    NTRIES = 0
    ANSWER = SUM + 1
    do while ANSWER ≠ SUM AND NTRIES ≠ 3
          NTRIES = NTRIES +1
          INPUT ANSWER
          if ANSWER = SUM then
              if NTRIES = 1 then FIRSTTRY = FIRSTTRY + 1
              CORRECT = CORRECT +1
              PRINT 'CORRECT'
          else if NTRIES ≠ 3 then
              PRINT 'WRONG. TRY AGAIN'
              else PRINT 'WRONG. LAST TRY'
          end
*
* CHECK IF STUDENT WISHES TO CONTINUE
* IF NOT, CLEAR CONTINUE FLAG
*
  PRINT 'DO YOU WISH TO CONTINUE?'
  INPUT ANSWER
  if ANSWER = 'N' then CONTINUE = 0
  end
*
* PRINT SCORE
*
PRINT 'SCORE IS', CORRECT, 'OUT OF', COUNT
PRINT 'INCLUDING', FIRSTTRY, 'THE FIRST TIME'
```

Fig. 8. This program allows the student three tries at the right answer.

A Loop Within a Loop

Give the student three chances at the problem.

In Fig. 8, we have expanded the program to allow the student three attempts at the right answer. Now the program contains several nested structures with loops inside loops. Note the increased number of cases that must be considered, e.g., whether a wrong answer is the third one and so must draw a different response than the 'WRONG. TRY AGAIN' given for the first two wrong answers. Note also that we must assign a wrong answer to ANSWER before entering the inside DO-WHILE loop. What would happen otherwise if a leftover value in ANSWER happened to be equal to SUM?

You may want to try expanding the program still further. Some of the features that you might want to add are:

- Telling the student whether a wrong answer is too large or too small.
- Asking the student whether he or she wishes to try again after a wrong answer.
- Allowing the student to determine the range of the numbers and to vary the range after completing a set of problems.

The logic of the program can become quite complex, but structured programming helps keep things manageable. The features that we have noted can make the program much more flexible and easier to use.

Disadvantages of Structured Programming

No treatment of a methodology is fair without some mention of its disadvantages. The disadvantages of structured programming are:

- It introduces an additional step in the program development process. The programmer must now write a structured program as well as the basic flowchart and the actual working program.
- Structured programming is difficult. The effort involved in writing a structured program may not be worthwhile if the actual program logic is simple. Of course, few interesting programs really have simple logic.
- The structured program does not usually result in a computer program that executes as quickly as possible or uses the least

memory. The structured program often requires extra variables, extra statements and awkward logic. Restricting program structures will make some tasks more difficult than they have to be.

- Structured programs require careful design and testing, particularly if the logic is complex.
- The structures may become confusing, particularly if there is nesting. Indenting and other formal procedures can help.
- The programmer must be consistent. The user's own language will be nonstandard and may be confusing and inconsistent. A programming language will be more standard and better defined, but will be difficult to check for errors without a compiler. Obviously, the simplest solution is to use a language that has structures; however, no such language is yet widely available for hobbyists.

Conclusion

Structured programming is a new technique that you should examine carefully. It brings discipline to the logic of programming and allows the user to check programs by hand before coding them for the computer. Structured programming is not easy, but it can result in fewer errors, greater confidence in programs, better documentation and faster program development.

David and Katie Bulman of Pragmatics convinced me of the importance of structured design, and Karl Amatneek of KVA Associates encouraged this writing. The examples were inspired by the work of Bob Albrecht and Don Inman in *Calculators/Computers*, their new magazine. ■

References

Glen Charnock, "Structured BASIC is Better!" *Kilobaud*, May 1977, pp. 104-106.

John Craig, "Structured BASIC . . . a negative view," *Kilobaud*, January 1977, pp. 122-123.

J. K. Hughes and J. I. Michtom, *A Structured Approach to Programming*, Prentice-Hall, Englewood Cliffs NJ, 1977.

Bill Jones, "Structured Programming," *Kilobaud*, January 1977, pp. 92-94.

C. L. McGowan and J. R. Kelly, *Top-Down Structured Programming Techniques*, Petrocelli-Charter, NY, 1975.

R. Ulrickson, "Software Modules are the Building Blocks of Structured Programming," *Electronic Design*, February 1, 1977, pp. 62-66.

E. Yourdon, *Techniques of Program Structure and Design*, Prentice-Hall, Englewood Cliffs NJ, 1975.

Source Listing the Hard Way

disassembling BASIC with BASIC

Clint H. Woeltjen
La Habra CA

This is meant to be a "how to do it" article for the creation of an assembly source listing of your 8080 BASIC interpreter. There are two programs involved, one for creating an octal and mnemonic listing of the interpreter (see Fig. 1), and the other for searching for specific instructions (see Fig. 9). They are written in Mits BASIC rev. 3.1, and with minor modification should be usable with practically any other 8080 BASIC. Using the flowcharts provided (see Figs. 2 and 10), along with the programs, you could even write a disassembler for 6800 BASIC. (Of course you would have to change the data statements to reflect the changed mnemonics.)

Using the technique described here, without the benefit of the mnemonic listing part of the program, it took me about twenty hours to complete the entire assembly listing. The addition of the mnemonic look-up part of the program should speed up the process considerably. I used an octal listing, as I am more familiar with octal, but both programs could be revised to use hexadecimal.

Getting A Source Listing of Your BASIC

The first thing is to produce the listing of your interpreter. You can use the program in Fig. 1 to output one page, or the entire 94 pages that comprise 8K BASIC. (Mits BASIC rev. 4.0 will be somewhat longer.) You should remember, however, that each page takes around six minutes to print, at least on my Teletype (a line printer is bound to be faster), so the entire 94 pages will take more than nine hours. Each page consists of 64 consecutive address locations, followed by three blank lines. When cut apart, they will fit nicely in a standard three-ring binder. Fig. 3 describes the

Fig. 1. 8080 BASIC disassembler.

```
1 DIM B(16)
10 REM "REMEMBER, ALLOW 6 MINUTES PER PAGE."
20 INPUT "STARTING ADDRESS, NUMBER OF PAGES.";P,Q
21 REM "THIS SETS UP A STARTING PLACE FOR THE DISASSEMBLER."
30 H$="01234567 "
40 P=INT(P/64)*64
41 REM "ALLOWS DISASSEMBLER TO START AT PAGE NEAREST TO"
42 REM "REQUESTED PAGE. PAGES CONSIST OF 64 LINES BEGINNING"
43 REM "WITH ADDRESS 000 000 ON PAGE 1."
50 FOR E=0 TO Q-1
51 REM "PAGE COUNTER."
60 FOR F=0 TO 63
61 REM "LINE COUNTER."
70 X=64*E+F+P
71 REM "ADDRESS OF CURRENT DATA BYTE BEING DISASSEMBLED."
80 A=PEEK(X)
90 C(0)=INT(X/256)
```

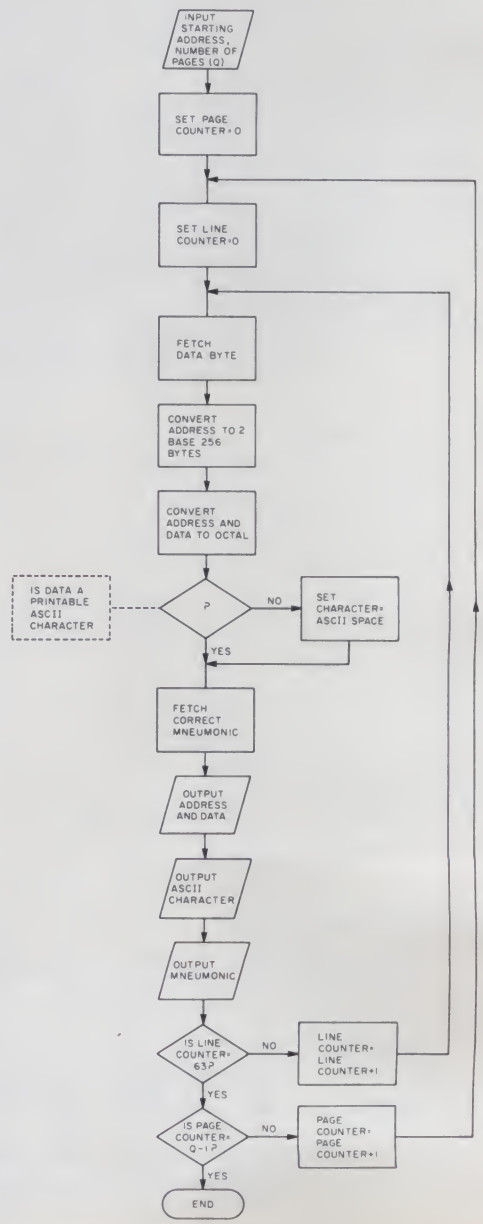


Fig. 2. Flowchart of the BASIC disassembler.

format of the list. Each line consists of two address bytes in octal, followed by one data byte. The ASCII equivalent of the data is presented next along with the required mnemonic.

Interpretation

When your listing is complete, begin interpreting the listing at address 000 000$_8$ (see Fig. 4). When you encounter a JMP instruction, jump to that location. For conditional jumps, and calls, write down on a separate sheet of paper the address *from* which you are jumping, and continue at the address specified by that instruction (see Fig. 5). This is basically what your computer does to keep track of calls and is invaluable in decoding the program. With multibyte instructions, cross out the mnemonics associated with the data or addresses, as they will only serve to confuse the issue (see Fig. 4).

When you reach a Return (RET), a Load Prog. Cntr (PCHL), or part of the program you have already decoded, return to the last address on the list, cross it out, and begin decoding from that location. (I found that it was nearly impossible to keep track of the data in all of the registers all of the time, so I was never quite certain to which location a PCHL was referring.) If you have to stop working on decoding (few people can keep going for twenty hours at a stretch without at least a few breaks), write down the last address you decoded so you will know where to begin again. When decoding, you should also note all LDAs, STAs, LHLDs, and SHLDs at the locations they reference, as these are usually data areas. This takes care of more than half of BASIC. The rest is accessed indirectly through PCHLs and will be discussed later.

Next use the ASCII characters to locate the symbol tables. Note that the program

```
91 REM "THIS ROUTINE CHANGES THE ADDRESS OF THE CURRENT"
92 REM "DATA BYTE INTO TWO BASE 256 NUMBERS."
100 C(1)=X-C(0)*256
110 C(2)=A
120 L=0
130 FOR I=0 TO 16
131 REM "THIS INITIALIZES THE OUTPUT STRING PLACE HOLDER"
132 REM "TO ALL BLANK SPACES."
140 B(I)=8
150 NEXT I
160 Y=A
161 REM "THIS ROUTINE CHECKS TO SEE IF THE DATA BYTE IS A"
162 REM "PRINTABLE ASCII CHARACTER. IT SUBTRACTS 128 FROM"
163 REM "BYTES GREATER THAN OR EQUAL TO 128 AS MITS BASIC"
164 REM "ADDS 128 TO THE LAST BYTE IN A SYMBOL IN"
165 REM "ORDER TO IDENTIFY THE END OF THE SYMBOL. ALL NON-"
166 REM "PRINT CHARACTERS ARE CHANGED TO SPACES."
170 IF Y>=128 THEN Y=Y-128
180 IF Y<=32 OR Y>=91 THEN Y=32
190 FOR I=0 TO 2
191 REM "STATEMENTS 190-290 CONVERT DATA AND ADDRESSES TO"
192 REM "OCTAL AND PLACE IN OUTPUT STRING HOLDER. THIS
193 REM "PERMITS PRINTING LEADING ZEROS."
200 X=C(I)
210 FOR J=0 TO 2
220 K=L+3*I+J
230 N=2-J
240 B(K)=INT(X/(8↑N))
250 X=X-(8↑N)*B(K)
260 NEXT J
270 L=L+3
280 IF K<6 THEN L=L-2
290 NEXT I
300 AA=INT(A/8)
301 REM "THIS LOCATES THE MNEMONIC EQUIVALENT OF THE CURRENT"
302 REM "DATA BYTE."
310 AB=A-AA*8
320 FOR J=0 TO AA
330 READ B$
340 NEXT J
350 B$=MID$(B$,AB*6+1,6)
360 RESTORE
370 FOR I=0 TO 16
371 REM "THIS PRINTS THE STRING EQUIVALENT OF THE DATA IN THE"
372 REM "OUTPUT STRING HOLDER."
380 A$=MID$(H$,B(I)+1,1)
390 IF I=16 THEN A$=CHR$(Y)
400 PRINT A$;
401 REM "THIS PRINTS THE ASCII EQUIVALENT OF THE CURRENT DATA"
402 REM "BYTE."
410 NEXT I
420 PRINT "     ";B$
421 REM "THIS PRINTS THE 8080 MNEMONIC EQUIVALENT OF THE"
422 REM "CURRENT DATA BYTE."
430 NEXT F
440 PRINT:PRINT:PRINT
450 NEXT E
1000 DATA "NOP   LXI B STAX BINX B INR B DCR B MVI B RLC   "
1001 DATA "---   DAD B LDAX BDCX B INR C DCR C MVI C RRC   "
1002 DATA "---   LXI D STAX DINX D INR D DCR D MVI D RAL   "
1003 DATA "---   DAD D LDAX DDCX D INR E DCR E MVI E RAR   "
1004 DATA "---   LXI H SHLD  INX H INR H DCR H MVI H DAA   "
1005 DATA "---   DAD H LHLD  DCX H INR L DCR L MVI L CMA   "
1006 DATA "---   LXI SPSTA   INX SPINR M DCR M MVI M STC   "
1007 DATA "---   DAD SPLDA   DCX SPINR A DCR A MVI A CMC   "
1008 DATA "MOV BBMOV BCMOV BDMOV BEMOV BHMOV BLMOV BMMOV BA"
1009 DATA "MOV CBMOV CCMOV CDMOV CEMOV CHMOV CLMOV CMMOV CA"
1010 DATA "MOV DBMOV DCMOV DDMOV DEMOV DHMOV DLMOV DMMOV DA"
1011 DATA "MOV EBMOV ECMOV EDMOV EEMOV EHMOV ELMOV EMMOV EA"
1012 DATA "MOV HBMOV HCMOV HDMOV HEMOV HHMOV HLMOV HMMOV HA"
1013 DATA "MOV LBMOV LCMOV LDMOV LEMOV LHMOV LLMOV LMMOV LA"
1014 DATA "MOV MBMOV MCMOV MDMOV MEMOV MHMOV MLMOV MMMOV MA"
1015 DATA "MOV ABMOV ACMOV ADMOV AEMOV AHMOV ALMOV AMMOV AA"
1016 DATA "ADD B ADD C ADD D ADD E ADD H ADD L ADD M ADD A "
1017 DATA "ADC B ADC C ADC D ADC E ADC H ADC L ADC M ADC A "
1018 DATA 'SSUB B SUB C SUB D SUB E SUB H SUB L SUB M SUB A "
1019 DATA "SBB B SBB C SBB D SBB E SBB H SBB L SBB M SBB A "
1020 DATA "ANA B ANA C ANA D ANA E ANA H ANA L ANA M ANA A "
1021 DATA "XRA B XRA C XRA D XRA E XRA H XRA L XRA M XRA A "
1022 DATA "ORA B ORA C ORA D ORA E ORA H ORA L ORA M ORA A "
1023 DATA "CMP B CMP C CMP D CMP E CMP H CMP L CMP M CMP A "
1024 DATA "RNZ   POP B JNZ   JMP   CNZ   PUSH BADI   RST 0 "
1025 DATA "RZ    RET   JZ    ---   CZ    CALL ACI   RST 1 "
1026 DATA "RNC   POP D JNC   OUT   CNC   PUSH DSUI   RST 2 "
1027 DATA "RC    ---   JC    IN    CC    ---   SBI   RST 3 "
1028 DATA "RPO   POP H JPO   XTHL  CPO   PUSH HANI   RST 4 "
1029 DATA "RPE   PCHL  JPE   XCHG  CPE   ---   XRI   RST 5 "
1030 DATA "RP    POP SWJP    DI    CP    PUSHSWORI   RST 6 "
1031 DATA "RM    SPHL  JM    EI    CM    ---   CPI   RST 7 "
OK
```

Address	Data	ASCII	Mnemonic
000 000	363		DI
000 001	303	C	JMP
000 002	375		- - -
000 003	002		STAX B
000 004	111	I	MOV CC
000 005	006		MVI B
000 006	234		SBB H
000 007	014		INR C

Fig. 3. Format of the output of the disassembler program.

000 000	363		DI
000 001	303	~~C~~	JMP (002 375)
000 002	375		- - -
000 003	002		~~STAX B~~
000 004	111	~~I~~	MOV CC
000 005	006		MVI B (234)
000 006	234		~~SBB H~~
000 007	014		INR C

You should continue at this address before processing address 000 004.

Fig. 4. You should begin interpretation in this manner. Cross out unneeded ASCII characters and mnemonics.

002 266	315	~~M~~	CALL (010 123)
002 267	123	~~S~~	~~MOV DE~~
002 270	010		- - -
010 123	066	~~6~~	MVI M (000)
010 124	000		~~NOP~~
010 125	315	~~M~~	~~CALL~~ (007 330)
010 126	330	~~X~~	~~RC~~
010 127	007		~~RLC~~
007 330	302	~~B~~	JNZ (004 233)
007 331	233		~~SBB E~~
007 332	004		~~INR B~~
004 233	314	~~L~~	CZ (011 000)
004 234	000		~~NOP~~
004 235	011		~~DADB~~
011 000	043		INX H
011 001	311	~~I~~	RET
004 236	311	~~I~~	RET
007 333	311	~~I~~	RET
010 130	315	~~M~~	CALL (011 333)
010 131	333		~~IN~~
010 132	011		~~DAD B~~

"Stack"
002 266
~~010 125~~
~~007 330~~
~~004 233~~
010 130

Fig. 5. An example of calls and jumps. This "paper stack" method will help keep you from becoming lost in a maze of conditional statements. It's also just about what the computer does when processing the program.

002 164	102	B	~~MOV BD~~
002 165	122	R	~~MOV DD~~
002 166	105	E	~~MOV BL~~
002 167	101	A	~~MOV BC~~
002 170	313	K	- - -
002 171	000		NOP
002 172	041	~~!~~	LXI H (000 004)
002 173	004		~~INR B~~
002 174	000		~~NOP~~
002 175	071	~~9~~	DAD SW

Symbol Table.

Note that the last character in the symbol equals the ASCII equivalent of the character + 200_8.

Fig. 6. Symbol tables can easily be identified using the ASCII characters.

002 320	036	MVI E (002)
002 321	002	~~STAX B~~
002 322	001	LXI B (024 036)
002 323	036	(MVI E) (024)
002 324	024	~~INR D~~
002 325	001	LXI B (000 036)
002 326	036	~~MVI E~~
002 327	000	~~NOP~~

Active on CALL (002 323).

Fig. 7. During normal execution the MVIE (024) is masked by the LXI B (024 036), which is rendered inoperable by the LXIB (000 036) which follows. However, upon being called, the MVI E (024) comes into execution.

004 315	333		IN (020)
004 316	020		- - -
004 317	346		ANI (002)
004 320	002		~~STAX B~~
004 321	312	~~J~~	JZ (004 315) (MPOUT)
004 322	315	~~M~~	~~CALL~~
004 323	004		~~INR B~~
004 324	361		POP SW
004 325	323	~~S~~	OUT (021)
004 326	021		~~LXI D~~

MPOUT
Label that you give routine to identify it.
Routine label at address pointed to by JZ instruction.

Fig. 8. Giving routines identifying labels consistent with their functions helps you remember their purpose later on.

subtracts 128_{10} from the data if its value is greater than 127_{10} (see Fig. 1). This is because Mits adds 128_{10} (200_8) to the last character of each symbol to signify its end (see Fig. 6). This might not be done with other versions of BASIC, but shouldn't cause problems even then. As you fill in the program, cross out the characters that are not part of a symbol table. Note the character strings that make sense and identify them in the margin. Cross out the mnemonics associated with the identified symbol tables (see Fig. 6).

Simple programs such as this which produce an assembly listing for each byte in memory can lead to misconceptions as to what is a routine, stack, or data. Be careful! Make sure that what you are crossing out should be crossed out, but also make sure that you cross out all that you can.

More complex programs that don't decode the one or two bytes following a multi-byte instruction can cause even more trouble when encountering hidden instructions such as those found in Mits BASIC. Fig. 7 shows an example of such an instruction. The first LXI B instruction serves no other purpose than to mask the MVI E instruction during normal routine execution. When the routine is called at the MVI E instruction, however, the LXI B instruction is rendered inoperative. Hidden instructions such as this are encountered several times in Mits BASIC, and should be watched for. Unless you wish to write a full scale disassembler, which probably would be nearly as complex as BASIC itself (and take as much memory), this technique of list, search, and decode is much easier.

OK, now you have the job more than half done. The second part of BASIC is accessed by the interpreter indirectly through the use of the PCHL instruction. As it is impractical to keep track of

the H and L registers all of the time, it is quite difficult to figure just where a PCHL is jumping. My solution was simply to ignore the PCHL instruction, and to look at the data as yet unprocessed. A series of bytes whose second byte is always less than 37_8 is a good indication that it is being used as a stack, or to store the H and L registers, and that it is not part of the routine.

Mits BASIC resides in memory below address 037 175_8, so addresses stored in the stack will usually be less than that. A few dozen 000s indicate either unused stack space, or unused data storage. When you reach a string of unprocessed bytes that is not obviously part of the stack, part of a symbol table, or data accessed by another part of the program, then proceed with decoding it, following the method outlined earlier.

Usually if you run across a jump or a call to an address that is beyond the amount of memory you have available in your computer, you know that you are not in a routine, or that you are out of sync with the routine. I ran across only one instance in which this wasn't true, a JP 176 341_8 which was either an error on my tape, or more probably another masking instruction which would normally never occur during program execution.

Interpreting the I/O

Now comes the fun part. You probably identified the input and output routines while disassembling, but what about those routines which access them? And what about all the rest? The program in Fig. 9 is designed to aid in the location of instructions which access other routines. (Keep an octal ASCII list handy, as this will help in identifying routines that check for spaces, carriage returns, line feeds, and other important ASCII symbols. If you see a MVI 012_8 followed by a Restart (RST), which accesses the output routine, you know that a line feed will be output to your terminal when that routine is accessed.)

First identify what the RST routines do. They are the routines that tie the entire program together. Follow them through until you reach a RET, or a loop whose only exit is a conditional return. Describe what the routine does. It may not be clear now as to why the routine wants to do that, but after seeing several instances as to when that RST is called, it will be easier to understand.

In my copy of BASIC, one of the RST routines is used to access the output routine, which makes it a lot easier to locate those routines which have something to output. Once you know what their purposes are (or at least what they do), have the search program search the interpreter for instances of each RST. Compare this list location for location with the listing you have made of BASIC, and cross out those instances where the RST instruction is really data or part of an address. Then start working through each routine which contains an RST and try to determine how it fits. This will offer a clue as to what the routine itself is trying to do.

Keep track of the contents of registers within a routine if possible. Does the routine manipulate data, addresses, or the contents of registers? Does it prepare a character for output, or request input? Where does it store important addresses and data? (And later, what other routines access this data?) This is particularly important for the output RST, as it is the major source of access to the output routine, and offers a clue as to the nature of many other routines.

When you have identified the purpose of a routine, give it a label consistent with its purpose, i.e., Main Program Output Routine could be

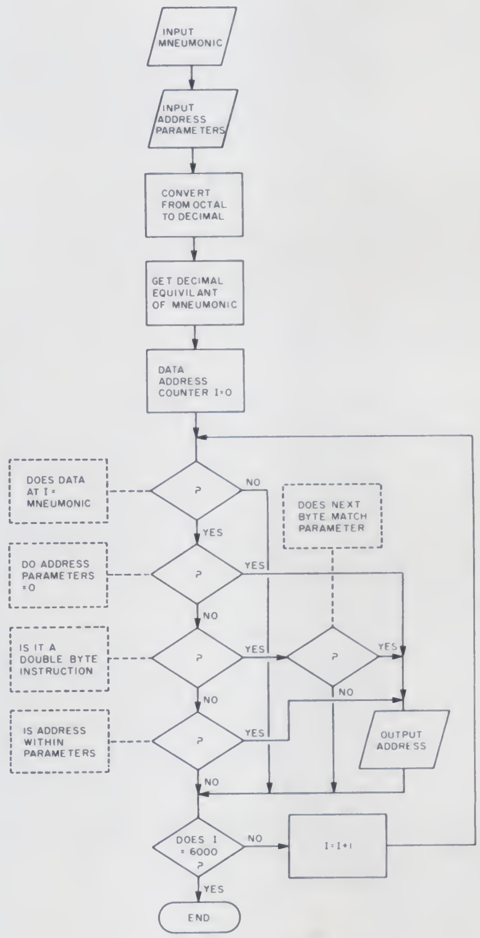


Fig. 10. Flowchart of Search Program.

Label	Begin	End
MPOUT	004 315	004 327
MPIN	004 330	004 355
INECHO	004 257	004 314
TLINE	004 246	004 256
CPUN	004 210	004 245

Fig. 11. Make a list of routine labels as each routine is identified. List the label, the beginning, and the ending addresses of the routine.

010 110	315	~~M~~	CAll (004 317) (MPOUT +2)	SOUT
010 111	317	~~O~~	~~RST 1~~	
010 112	004		~~INR B~~	
010 113	043	#	INX H	
010 114	005		DCR B	
010 115	302	~~B~~	JNZ (010 110) (SOUT)	
010 116	110	~~H~~	~~MOV CB~~	
010 117	010		- - -	

Fig. 12. When calling a routine in the middle, show the routine name plus the number of bytes past the beginning.

called MPOUT (see Fig. 8). Write down in a list the routine label, and the beginning and ending addresses of the routine (see Fig. 11). This is useful later if you want to locate a specific routine without having to search all of BASIC again for it. Then have the search program find all calls and jumps to within that routine.

Suppose you found a CALL that accesses MPOUT at two bytes past the initial byte of the routine. You would write down after the CALL, MPOUT +2 (see Fig. 12). Before you continue to the next call or jump, try to identify the purpose of the routine in which the call is located. Label it and continue from where you left off. Use the same procedure you used earlier in making a paper stack, as this will prevent routines and calls from being missed.

Eventually, with a lot of patience and a bit of work, you will have an assembly source listing of your BASIC interpreter. If you really want to go into detail, you could at this point make a flow diagram showing each routine as a block and locating all entry and exit addresses. This might make it easier to patch, but is not a necessity.

Fig. 9. 8080 BASIC Search Program.

```
1 DIM A(3),B(3),C(1)
10 INPUT "INSTRUCTION MNEMONIC";A$
11 REM "INPUT MNEMONIC AS A 6 BYTE WORD. YOU MUST INCLUDE"
12 REM "SPACES OR IT WILL NOT WORK."
20 INPUT "LOW, HIGH ADDRESSES IN OCTAL.";A(0),A(1),A(2),A(3)
21 REM "THIS ROUTINE INPUTS MNEMONIC AND ADDRESS PARAMETERS"
22 REM "TO BE SEARCHED FOR."
30 FOR I=0 TO 3
31 REM "CHANGES ADDRESS FROM OCTAL TO BASE 256."
40 X=A(I)
50 GOSUB 300
51 REM "OCTAL CONVERSION ROUTINE."
60 B(I)=Y
70 NEXT I
80 B(0)=B(0)*256+B(1)
81 REM "CHANGES ADDRESSES FROM BASE 256 TO DECIMAL."
90 B(1)=B(2)*256+B(3)
100 FOR I=0 TO 31
105 READ B$
106 REM "RETRIEVES A LINE OF 8 MNEMONICS."
110 FOR J=0 TO 7
115 C$=MID$(B$,J*6+1,6)
116 REM "SEPARATES ONE MNEMONIC FROM LINE."
120 IF A$=C$ THEN B(2)=I*8+J
121 REM "SETS B(2) EQUAL TO THE DECIMAL EQUIVALENT OF THE"
122 REM "INSTRUCTION MNEMONIC, IF THE INSTRUCTION MNEMONIC"
123 REM "MATCHES THE MNEMONIC RETRIEVED."
125 NEXT J
130 NEXT I
140 RESTORE
150 FOR I=0 TO 6000
160 X=PEEK(I)
161 REM "SEARCHES BASIC FOR INSTANCES OF INSTRUCTION REQUESTED."
170 IF X<>B(2) THEN 260
175 IF B(0)=0 AND B(1)=0 THEN 220
176 REM "TESTS FOR A SINGLE BYTE INSTRUCTION."
180 X=PEEK(I+1)+256*PEEK(I+2)
190 IF INT(B(0)/256)=B(0)/256 AND B(1)=0 THEN 240
191 REM "TESTS FOR A DOUBLE BYTE INSTRUCTION."
200 IF X<B(0) OR X>B(1) THEN 260
201 REM "CHECKS FOR ADDRESS WITHIN GIVEN LIMITS."
210 IF X>B(1) THEN 260
220 GOSUB 370
221 REM "OUTPUT ROUTINE."
230 GOTO 260
240 X=PEEK(I+1)
250 IF X=B(0)/256 THEN 220
251 REM "TESTS FOR CORRECT DOUBLE BYTE INSTRUCTION."
260 NEXT I
270 END
300 Y=0
310 FOR J=0 TO 2
320 N=2-J
330 Y=INT(X/(10↑N))*INT(8↑N)+Y
340 X=X-INT(X/(10↑N))*INT(10↑N)
350 NEXT J
360 RETURN
370 C(0)=INT(I/256)
380 C(1)=I-C(0)*256
390 FOR J=0 TO 1
400 Y=0
410 FOR K=0 TO 2
420 N=2-K
430 Y=INT(C(J)/(8↑N))
440 C(J)=C(J)-INT(C(J)/(8↑N))*(8↑N)
450 Y$=RIGHT$(STR$(Y),1)
460 PRINT Y$;
470 NEXT K
480 PRINT " ";
490 NEXT J
500 PRINT
510 RETURN
1000 DATA "NOP   LXI B STAX BINX B INR B DCR B MVI B RLC   "
1001 DATA "---   DAD B LDAX BDCX B INR C DCR C MVI C RRC   "
1002 DATA "---   LXI D STAX DINX D INR D DCR D MVI D RAL   "
```

Summing Up

Now it is up to you as to what you want to do with the finished product. Myself, I want more than six digits of precision in some of my calculations, but I don't want the memory overhead of an extended BASIC. I could use a simple accounting BASIC without all of the trigonometric functions, but with some additional formatting of the output. I also don't relish the idea of writing an entire 6000 byte program from scratch. You could even create a special BASIC for each application you have in mind, releasing memory that would otherwise be tied up in routines that you aren't using.

One last note. Your bootstrap will locate the addresses of the interpreter's internal loader. In my copy of BASIC it resides in locations 037 000$_8$ thru 037 175$_8$. When BASIC is initialized and a program loaded, this routine is written over and lost. If you want a copy, you can toggle it from the front panel. Stop your computer when BASIC has finished loading. Examine the address pointed to by the bootstrap. Write down the data displayed by the data lights (if you have them), examine next, write down the data, and continue in this manner until you have copied the entire loader.

If you don't have front

panel LEDs, then you probably have some sort of output program on PROM. You could write a short machine language program that would have the output program list the loader. Without the PROM output program, such a machine language dump program could end up being more complex than the loader you are trying to make a copy of, and might not be worthwhile using.

When writing routines for BASIC, make sure that you don't overwrite any other routines, and make sure that you locate *all* points of entry to the routine. You could get some rather unexpected results if you don't. ■

```
1003 DATA "---    DAD D LDAX DDCX D INR E DCR E MVI E RAR   "
1004 DATA "---    LXI H SHLD   INX H INR H DCR H MVI H DAA   "
1005 DATA "---    DAD H LHLD   DCX H INR L DCR L MVI L CMA   "
1006 DATA "---    LXI SPSTA   INX SPINR M DCR M MVI M STC   "
1007 DATA "---    DAD SPLDA   DCX SPINR A DCR A MVI A CMC   "
1008 DATA "MOV BBMOV BCMOV BDMOV BEMOV BHMOV BLMOV BMMOV BA"
1009 DATA "MOV CBMOV CCMOV CDMOV CEMOV CHMOV CLMOV CMMOV CA"
1010 DATA "MOV DBMOV DCMOV DDMOV DEMOV DHMOV DLMOV DMMOV DA"
1011 DATA "MOV EBMOV ECMOV EDMOV EEMOV EHMOV ELMOV EMMOV EA"
1012 DATA "MOV HBMOV HCMOV HDMOV HEMOV HHMOV HLMOV HMMOV HA"
1013 DATA "MOV LBMOV LCMOV LDMOV LEMOV LHMOV LLMOV LMMOV LA"
1014 DATA "MOV MBMOV MCMOV MDMOV MEMOV MHMOV MLMOV MMMOV MA"
1015 DATA "MOV ABMOV ACMOV ADMOV AEMOV AHMOV ALMOV AMMOV AA"
1016 DATA "ADD B ADD C ADD D ADD E ADD H ADD L ADD M ADD A "
1017 DATA "ADC B ADC C ADC D ADC E ADC H ADC L ADC M ADC A "
1018 DATA "SUB B SUB C SUB D SUB E SUB H SUB L SUB M SUB A "
1019 DATA "SBB B SBB C SBB D SBB E SBB H SBB L SBB M SBB A "
1020 DATA "ANA B ANA C ANA D ANA E ANA H ANA L ANA M ANA A "
1021 DATA "XRA B XRA C XRA D XRA E XRA H XRA L XRA M XRA A "
1022 DATA "ORA B ORA C ORA D ORA E ORA H ORA L ORA M ORA A "
1023 DATA "CMP B CMP C CMP D CMP E CMP H CMP L CMP M CMP A "
1024 DATA "RNZ   POP B JNZ   JMP   CNZ   PUSH BADI   RST 0 "
1025 DATA "RZ    RET   JZ    ---   CZ    CALL   ACI   RST 1 "
1026 DATA 'RNC   POP D JNC   OUT   CNC   PUSH DSUI   RST 2 "
1027 DATA "RC    ---   JC    IN    CC    ---   SBI   RST 3 "
1028 DATA "RPO   POP H JPO   XTHL  CPO   PUSH HANI   RST 4 "'
1029 DATA "RPE   PCHL  JPE   XCHG  CPE   ---   XRI   RST 5 "
1030 DATA "RP    POP SWJP    DI    CP    PUSHSWORI   RST 6 "
1031 DATA "RM    SPHL  JM    EI    CM    ---   CPI   RST 7 "
OK
```

Happy Valentine's Day!

How Good Is Tarbell's Floppy Interface?

Part 1: Overview

The disk-oriented system.

(Photos by David Mochizuki.)

Larry Fukumoto, D.D.S.
5633 Lincoln Ave.
Cypress CA 90630

Once his computer system is functioning properly and the hobbyist gains some prowess in programming, he finds program development causes the computer to develop a voracious appetite for memory. The floppy-disk system with a good memory management system will eliminate the need for additional memory, and also act as storage for his valuable programs. The purpose of this article is to describe my experiences in building and interfacing a floppy-disk system to my computer. A future article will deal with how the card operates (hardware description) and the interrelation between hardware and software.

Rationale

The major obstacle to a floppy system is the cost. With the introduction of the Tarbell floppy-disk interface, the price of system implementation is about $900 with all-new equipment, and about $500 with used equipment. For comparison, assuming the cost of the floppy-disk system is $750 with memory cost at about $25 per kilobyte (8K prices), about 30K of memory can be purchased for the equivalent price. The floppy disk has about 240K capacity, is nonvolatile and can be increased with a change of a diskette. The trade-off is an evanescent speed differential.

Applications

Because of the prodigious increase in storage capacity, and the ease and speed in locating files, I am discovering numerous uses for the system such as text editing, patient records, accounts receivable and payable and many miscellaneous applications. I can develop large programs far exceeding the capacity of my computer, since the floppy-disk operating system writes out data to disk beyond the capacity of the computer, and reads in data from the disk when editing or assembling a large program. Writing this article was easy using the floppy-disk system and its text editor.

Description

Actually, the interface is a controller and interface combined. A soft-sectored disk system is utilized with a programmed data transfer rather than a DMA system. The interface includes a bootstrap program on ROM which is 32 bytes long. It is automatically enabled when reset is toggled on the computer, and switches itself out after the bootstrap program is run.

The operation of this ROM system is beyond the scope of this article, so it will be described in the hardware article to follow. The interface operates at about 250K bits per second. The soft-sector formatted diskette has a maximum capacity of about 240K bytes.

Construction

The interface unit comes in a small box; the printed circuit board is in a separate plastic bag. All other components are in smaller bags, with the FD 1771 chip in its own foil-insulated container. The 68-page manual is divided into chapters

on assembly instructions, system checkout, operating instructions, theory of operation, and a final section with pinouts of ICs used and the Western Digital specification sheets on the FD 1771.

The PC board is well designed, and the parts well laid out. The traces are essentially linear, so sighting horizontally down the component side and vertically down the solder side permits a quick visual check for any flaws (there were none). All ICs have pin 1 oriented in the same direction. The silkscreen mask is accurate, making component placement simple.

The instructions are designed to be read and followed carefully. They do not require placement and soldering of discrete components of a single class (locate and solder R22-R58; locate and solder C1-C85, etc.), but rather the assembly and testing of a single section at a time. The card can be assembled confidently, and correctly, one section at a time. This method yields better results than completing the card, checking a few voltages, and then doing a "smoke test" (I never fail a smoke test . . . it smokes every time!). Tarbell assumes you have a functioning computer mainframe with an S-100 bus and front panel. The card is plugged in after each section is completed, and tested by checking voltages on the various chips in that section, using a probe consisting of a 330 Ohm resistor attached to an LED supplied with the kit, and toggling the sense switches. If there is a problem, only the section being checked out needs troubleshooting. This kit is based on modular assembly.

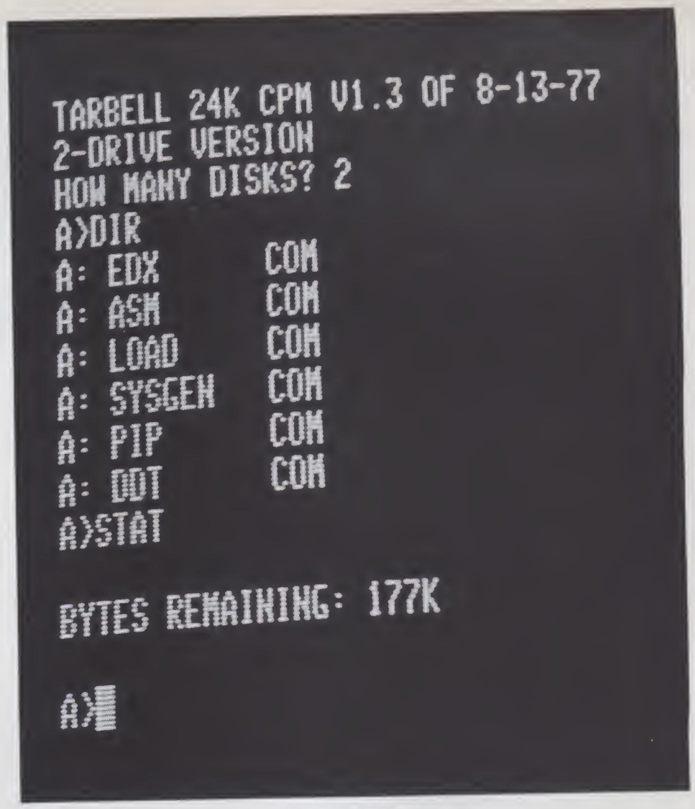
Although this is one of the more complicated cards I have built, it has proved to be the most educational and fun. As each section is put together (regulator, primary address, master reset, extended address, input/output) the tests show the relation of the interface to the hardware and, in steps encountered later, the software to the hardware.

Interfacing

After the interface card is completed, the interconnections to the disk drive are made via a ribbon cable. The manual describes all the pads and their functions relative to the drive, interface and computer. The many options available on the interface and disk drive give the impression of a variety of capabilities and configurations. This is true to the extent that this card can be configured to most drives currently on the market. Descriptions are given to interconnect the Innovex 410, Innovex 220, GSI 110 and Shugart 800/801. A single 50-pin connector and cable facilitate the interconnection. The configuration for the drive is quickly and firmly established, and easily implemented.

Operation

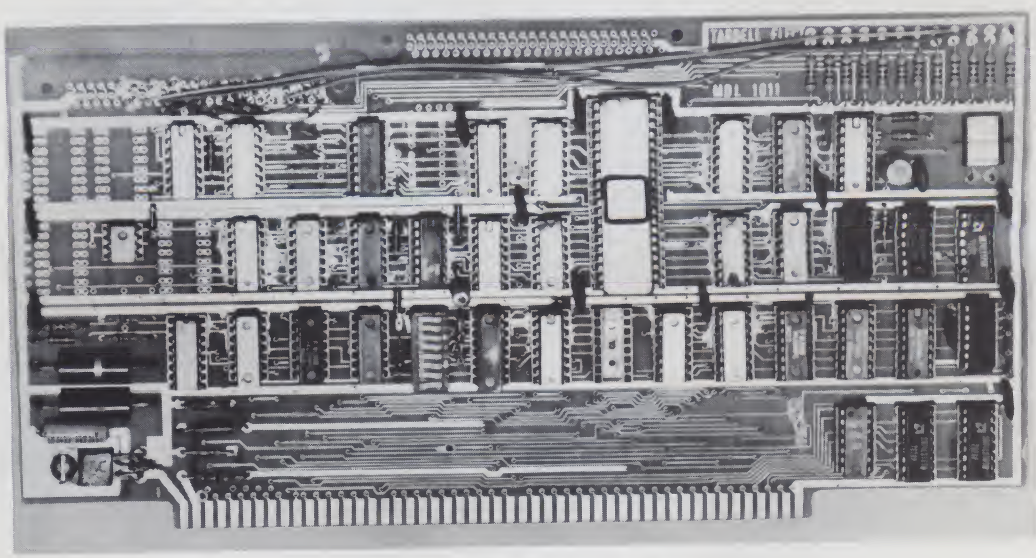
Next is a series of steps that test the control functions. The stepper motor is moved in, the drive turned on, and the reset switch is depressed. If the connections are correct, the head seeks track 0 when the switch is released. Next, a seven-byte program is toggled in to check single-step in and single-step out. Then, a short seek program is entered that allows various track numbers to be entered, and the drive is visually checked to see if it seeks different tracks on a qualitative basis. Each of these preliminary tests is done in a logical and progressive sequence.



It comes alive!

Finally, programs to write and read a sector on a track are toggled in and tested. When these tests are complete, the system is running.

I inserted the diskette and pressed system reset. With the CP/M software, the system booted itself in and displayed its title and prompt character A>. I typed DIR—the drive clicked a few times and the console screen filled with each of the files on the disk. As I played with the various routines on my new system, it suddenly occurred to me . . . my little hobby system was behaving like a real computer! ■



The Tarbell floppy interface.

Manipulating ASCII Data

packing and unpacking techniques

Jack Ward
Route 5, 901 Vernon Ave.
Fulton MO 65251

As soon as you have a basic microcomputer wonderful things start to happen. You can enter programs and data by the panel switches, and startle and amaze captive audiences of friends and relatives. Sooner or later though, resetting eight panel switches for each byte entered loses some of its thrill.

The next step might be to build a hexadecimal keyboard. This would reduce the eight switch settings to two button pressings, which is quite an improvement. This too, however, lacks the sophistication of keyboard entry, and each of us wants to upgrade his input device to a Teletype or video display unit (VDU) with keyboard.

Unless you are experienced or forewarned, this is where the first rude shock occurs. (You are about to be forewarned.) The switch entry or hex pad input is in binary. By this time you have probably gotten so used to converting denary (base 10) numbers to binary before entering them that the conversion seems perfectly natural. The keyboard input values are not binary; they are American Standard Code for Information Interchange (ASCII) coded. Arithmetic with ASCII representations will give weird and wonderful results.

ASCII Arithmetic Operations

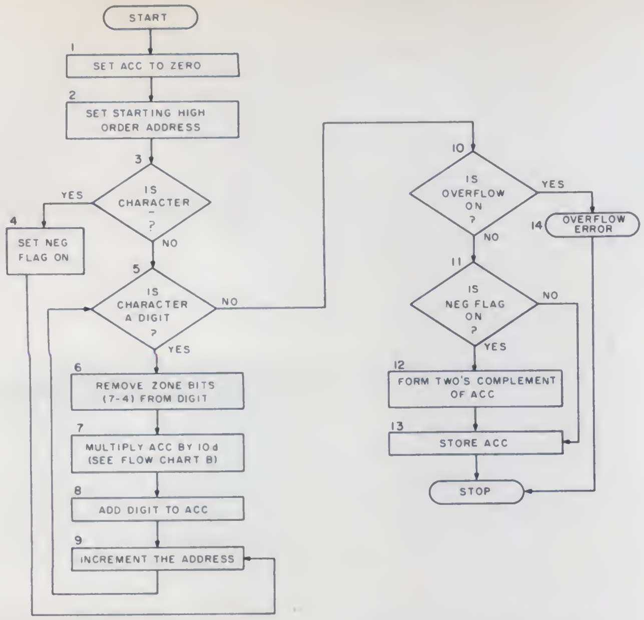
Let's look for a moment at Fig. 1. The denary digits are shown in the first column. Opposite them is (probably) what you will get from ASCII. I say probably because one of the bits (b7) is unused in coding the value and may be used for parity or some other purpose. My notation for the bits is as follows.

The leftmost bit is called b7, the rightmost bit is called b0, and they decrease in value from b7 in powers of two. When it is useful, I will use d for denary (base ten) and b for binary (base two).

Let's say that we enter two numbers, a 2 and a 3, and we want to add them together. The ASCII 2 (00110010) and 3 (00110011) give a binary sum of 01100101. From Fig. 1, this is clearly not an ASCII 5. In fact, if you can display it, this binary pattern will appear as a lowercase e.

Denary	ASCII Representation Zone Numeric 76543210
0	00110000
1	00110001
2	00110010
3	00110011
4	00110100
5	00110101
6	00110110
7	00110111
8	00111000
9	00111001
(minus) –	00101101

Fig. 1. ASCII representation.



Flowchart A. Packing ASCII characters into a binary word.

Numbers longer than one digit add to the confusion.

For binary arithmetic purposes, ASCII values must be converted to binary and packed into one word. A word may be one eight-bit byte or longer depending on your micro. We also have to recognize the limitations of word size, and to consider the implications of negative numbers which are generally stored in two's complement form.

Taking two's complement first, a negative number is stored so that adding the positive and negative forms of the same value results in a zero with a carry beyond b7. For example, 7d is 00000111b, and -7d is 11111001. The binary sum of these two values is 1 00000000. Happily, 11111001b works as -7d in all arithmetic operations. The process of forming the two's complement of a binary value is not difficult. It consists of first inverting all the ones and zeros (zeros become ones and ones become zeros), then adding one.

ASCII-to-Binary Conversion, Clarifying the Mechanics

Now for the conversion of ASCII values to binary. Flowchart A does this, but we may need to clarify some of the mechanics before programming it. Boxes 1 and 2 start by preparing a place to build our binary number, and locating the first character of the ASCII input value. I say character rather than digit because the first character could be a minus sign. The conversion process will alter the ASCII codes, so we would probably have moved the ASCII string to a work area before processing it. This leaves the original ASCII code available to be displayed if needed.

Box 3, a test, checks to see if the first character is a minus sign. This is done by comparing the character with an ASCII minus sign (00101101b). If it is a minus sign, set a byte (or bit), normally zero to one (box 4), and proceed to box 9 for the next character. The super-cautious may wish to insert a check for a leading plus, as well. (If a plus were found, we would just move to the next character.)

In any event, we will end up at box 5. Box 5 tests the character to determine whether or not it is an ASCII digit. That is, to see if it is equal to 00110000b or 00111001b or something in between. If it is an ASCII digit, we turn it into binary at box 6 by removing bits b7 through b4. This can be done by a logical AND with the character in the accumulator and a mask of 00001111b. Since an AND requires a one bit in both values to produce a one in the result, the four zeros can only produce zeros, while the four ones reproduce whatever is in bits b3 through b0 of the character. At box 7, we multiply the accumulator by 00001010b (10d). If you do not have hardware multiply, you will find the four instruction sequence in Flowchart B to be very efficient. The first time through the loop this multiplication produces a product of zero, but it will be required to process the other digits.

Box 8 adds our transformed digit to the accumulator. At box 9, we advance the data address to the next character. Now back to box 5 to see if we have another digit. If we do, we go around the loop again. Eventually, we will run out of digits and go to box 10 to see if we have gone beyond our allowable size. If a micro has only byte arithmetic, seven bits will be the maximum (one bit acts as the sign). Other computers may use larger words or have the facility of doing arithmetic on multiple bytes (extended precision).

If we have gone beyond the allowable length, we have an error condition and go to box 14 which sets an error flag. The use made of this flag is up to you.

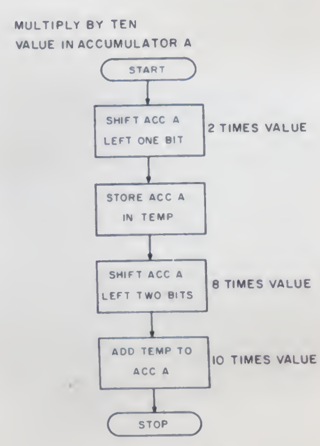
If we have a valid binary number, it is in its positive form. Box 11 checks to see if the negative flag is on. If it is, box 12 forms the two's complement in the accumulator. We now have our number with its proper sign, and box 13 stores this value at a predetermined address.

Let's take a look at the state diagram in Fig. 2. This shows the process of converting -123d from ASCII (00101101 00110001 00110010 00110011b) to a packed binary value. Each line shows the number of the flowchart box just completed and the new values of any changed data. The final line shows all of the data areas.

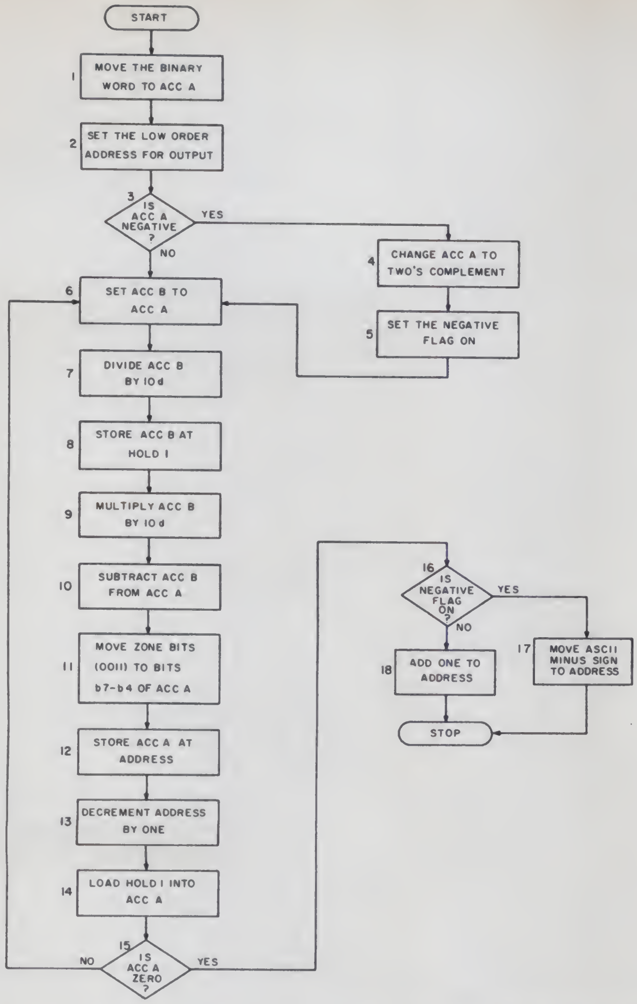
Now we have a binary number and as many more as we need (by the same method). We can do our arithmetic and examine the results. That is, we can examine the results if we want to use the binary display LEDs on the front panel. If, however, we would like to print the result or display it on a VDU, we have our original problem in reverse. We have a binary form and we need an ASCII representation. So on we go to Flowchart C.

Flowchart C assumes that there are two accumulators and one temporary storage byte available for use. This can be done with one accumulator and two temporary storage areas, but it involves much data movement between the accumulator and the storage areas.

The word containing the binary number is loaded into accumulator A in box 1, and box 2 sets the low order address for the ASCII output. The number of words needed for the output depends on the word size. An eight-bit word can store a signed value of approximately ±128d, and would need a four-byte area to record the number and sign. Similarly, a twelve-bit word would need a maximum of five bytes (±2048d), and a sixteen-bit word would need



Flowchart B.



Flowchart C. Unpacking a binary word into ASCII characters using two accumulators and a temporary storage area.

a maximum of six bytes (±32768d). It is a good idea to have the output area all ASCII spaces (00100000b) initially. If you are using an algorithm to link bytes together for extended precision arithmetic, you should adjust the size of the output area accordingly.

The test at box 3 determines the sign of the binary number. If it is negative, box 4 forms the two's complement in accumulator A, and box 5 sets a flag (normally off) to on. This flag will allow us to put a minus sign in the proper place in the output area. Box 6 moves a copy of our binary number (in its positive form) to accumulator B.

Accumulator B is divided by ten at box 7, and box 8 stores a copy of this quotient in a byte called HOLD1. Box 9 multiplies accumulator B by 10d. Accumulator B is subtracted from accumulator A at box 10, giving a binary digit between zero and nine in value.

Binary-to-ASCII Conversion

Zone bits to change the binary character to its ASCII representation are inserted in box 11. This is usually done by a logical (nonexclusive) OR on the accumulator with a mask of 00110000b. (This forces ones into positions b5 and b4). Box 12 stores the ASCII character that is in accumulator A at ADDRESS. The storage address is decremented by one at box 13.

Box 14 recovers the remainder of the binary value and stores it in accumulator A. This quantity is tested in box 15. If it is not zero, the process returns to box 6 to find the next ASCII digit.

If accumulator A is zero, box 16 checks to see if the negative flag is on. If the negative flag is on, box 17 moves an ASCII minus sign (00101101b) to ADDRESS. If the negative flag is off, the address is reduced by one at box 18. In both cases, address points to the first (high order) character in the ASCII string representing the original binary number.

The state diagram in Fig. 3 shows the process of converting -123d (10000101b) from binary to an ASCII string. Except for the first and last lines, only the values involved or changed are shown. The box numbers show the flow of the action through the flowchart.

Summary

Now you can convert from ASCII to packed binary and from packed binary to ASCII. It may be that you would like to be able to use values larger than those that can be stored in one word. This is possible, and I expect to discuss this at a later time. If you think that you would eventually like the capability of extended precision in your arithmetic, I suggest that the Add, Subtract, Multiply, and Divide instructions be done by subroutines, so that they can be easily replaced later on. For now, smile and feel sophisticated computing in binary and displaying in ASCII. ■

BOX	Input Area A	A+1	A+2	A+3	ACC	Addr	NEG Flag
	00101101	00110001	00110010	00110011	????????	?	0
1					00000000		
2						A	
4							1
9						A+1	
6		00000001					
7					00000000		
8					00000001		
9						A+2	
6			00000010				
7					00001010		
8					00001100		
9						A+3	
6				00000011			
7					01111000		
8					01111011		
9						A+4	
12					10000101		
13	00101101	00000001	00000010	00000011	10000101	A+4	1

Fig. 2. State diagram using Flowchart A to pack an ASCII numeric string into a binary word.

QUESTIONNAIRE

Are you interested in microcomputers for:

- ☐ a hobby?
- ☐ your business?
- ☐ other?________________

Have you a microcomputer system yet?________

If so, please describe make, memory, peripherals.

____

____

Approximate investment so far $

What articles would you like to see in Kilobaud?

____

____

____

Place
9 ¢
Stamp
Here

kilobaud

PETERBOROUGH NH 03458

BOX	ACC A	ACC B	HOLD1	A	A+1	A+2	A+3	Addr	NEG Flag
				Output Area					
1	10000101	????????	????????	00100000	00100000	00100000	00100000	?	0
2								A+3	
4	01111011								
5									1
6		01111011							
7		00001100							
8			00001100						
9		01111000							
10	00000011								
11	00110011								
12							00110011		
13								A+2	
14	00001100								
6		00001100							
7		00000001							
8			00000001						
9		00001010							
10	00000010								
11	00110010								
12						00110010			
13								A+1	
14	00000001								
6		00000001							
7		00000000							
8			00000000						
9		00000000							
10	00000001								
11	00110001								
12					00110001				
13								A	
14	00000000								
17	00000000	00000000	00000000	00101101	00110001	00110010	00110011	A	1
	Display character				1	2	3		

Fig. 3. State diagram using Flowchart C to unpack a binary value into an ASCII string.

Read any Good Books Lately?

a program to test readability

Al Gerbens
1038 E. 6th Pl.
Mesa AZ 85203

At what reading level do you write? Thinking about putting together an article for *73, Kilobaud* or (heaven forbid) some other magazine? Writing a term paper for that night school course? Use this program to evaluate the content of your first draft, and you'll find yourself using a lot more plain English. The result could be easier reading and better understanding of the material.

The Art of Readable Writing by R. Flesch is a book that outlines a method for the objective evaluation of written material. The readability of the text is broken down into two categories: 1. Ease of understanding. 2. Human interest content. The first is determined using the average number of syllables per word and the average number of words per sentence. The second is based on the number of "personal" words and sentences contained in the text being tested.

Analyzing written material seemed like a challenging job for my microcomputer so I set about trying to transfer Flesch's concepts into a basic computer program. A few compromises were made, but I think the result is a useful tool for objective categorization as well as an interesting application for a home microcomputer system.

Program Description

The text sample size is about 50 words. The program will reject an attempt to enter a string length less then 120 on the basis of insufficient sample size. Change lines 70 and 72 to delete for debugging or changing the reject level.

A more accurate evaluation can be obtained by increasing the sample size, and the easiest way is to run the program twice and average the results. An alternative is to extend the number of string entries and change the string summation statement in line 68. My system limits the string length, which can be entered using a single input statement to about 140; yours may not have this limitation. The program first counts the number of words in string E$ (defined in line 68) by counting the number of blank spaces in the string. It then discounts the second blank space following the end of a sentence. It also checks to make sure that the space is not a leading space at the beginning of the material. In determining if a blank space is preceded by another blank space, the number of sentences are determined. Next, the program begins the task of determining the number of "personal" words in E$. These words are found in the strings P$, H$, R$, D$, I$ and T$ (defined in lines 48 through 58). These strings consist of personal pronouns and gender-related words grouped by word length. For example, P$ is a string of three-letter words. Each

```
THE NUMBER OF SYLLABLES HAS NOW BEEN DETERMINED.
```

Example 1.

Program listing.

```
10 #"        READABILITY TEST"
12 FOR X=1TO10:#"":NEXT:FORX=1TO1000:NEXT
14 #"THIS IS A TEST OF READABILITY"
16 #"BASED ON EASE OF UNDERSTANDING"
18 #"AND HUMAN INTEREST CONTENT."
20 #"SELECT ABOUT 50 WORDS OF TEXT"
22 #"AND ENTER IT AS REQUESTED."
24 FORX=1TO2000:NEXT
26 #"USE PARENTHESIS INSTEAD OF"
28 #"QUOTATION MARKS."
30 FORX=1TO2000:NEXT
32 #"PLEASE ENCLOSE ALL PROPER NAMES"
34 #"OF PERSONS IN PARENTHESIS ALSO."
36 DIMH$(150)
38 DIMX$(150)
```

string is expandable by simply redefining it. Two precautions should be observed when doing this: 1. Be sure you add the word to the correct string according to word length. 2. Check the appropriate dimension statement in lines 36 through 47 for adequate reserved space.

Basically, the program sifts through E$ and one of these strings of personal words checking for substring identity. When identity occurs, a check is made for a blank space on both sides. The check is discounted for first words and ends of sentences. Part of this job is handled by subroutine 194.

Next, the number of vowels is counted using the loop in line 136 to 148. An assumption is made that the number of syllables is equal to 90 percent of the number of vowels. (A routine to actually count syllables is beyond me and the size of my machine.) The number of quotes, question marks and exclamation marks are then counted. Using this information, and equations located in lines 234 and 240, a number between 0 and 100 is generated for reading ease and human interest. These numbers are then interpreted using the appropriate statements relating to difficulty, level and interest.

Program Comments

The program takes three or four minutes to execute, and without a front panel of blinking lights you can easily suspect something is amiss while your computer cranks. An attempt to break this up is included using statements indicating the end of each task. See Example 1.

The program takes about 4.7K of RAM, not counting the interpreter, which could be reduced considerably by trimming the dimension statements. Also, rewriting the personal word check and

```
READABILITY TEST
THIS IS A TEST OF READABILITY
BASED ON EASE OF UNDERSTANDING
AND HUMAN INTEREST CONTENT.
SELECT ABOUT 50 WORDS OF TEXT
AND ENTER IT AS REQUESTED.
USE PARENTHESIS INSTEAD OF
QUOTATION MARKS.
PLEASE ENCLOSE ALL PROPER NAMES
OF PERSONS IN PARENTHESIS ALSO.

ENTER ABOUT 4 LINES OF TEXT
?IN ORDER FOR US TO UNDERSTAND M
ORE ABOUT COMPUTERS, I AM TRYING
TO GET THE MANUFACTURERS TO WRI
TE ARTICLES TELLING US ABOUT THE
IR
ENTER NEXT 4 LINES OF TEXT
?SYSTEMS — WHY THEY WERE DESIGNE
D THE WAY THEY WERE — WHAT THE B
ENEFITS ARE TO US, THE USERS, AN
D WHAT THE TRADE-OFFS WERE.
THE NUMBER OF WORDS HAVE NOW
BEEN COUNTED.

THE NUMBER OF SENTENCES HAVE
BEEN DETERMINED.

THE NUMBER OF SYLLABLES
HAVE NOW BEEN COUNTED.

THE NUMBER OF PERSONAL WORDS
HAVE BEEN COUNTED.

READING EASE = 38.68
HUMAN INTEREST = 30.93
THE READING EASE SCORE OF
38.6
INDICATES THE MATERIAL IS
DIFFICULT TO READ; HIGH SCHOOL-JR.
COLLEGE LEVEL.
THE HUMAN INTEREST SCORE OF
30.9
INDICATES THAT THE MATERIAL IS
INTERESTING; SIMILAR TO MATERIAL
FOUND IN DIGEST TYPE PUBLICATIONS.
```

Sample run.

```
40 DIMQ$(150)
42 DIMD$(100)
44 DIME$(325)
46 DIMT$(100)
47 DIMP$(50)
48 P$="YOUHIMHERSHEMANMENBOYSONHIS"
50 H$="SISTERMOTHERFATHERNEPHEWTHEIRS"
52 R$="MEHEMY"
54 D$="UNCLETHEIRFOLKSYOURS"
56 I$="I"
58 T$="THEYTHEMGIRLAUNTWIFEYOURMINEHERSSONSBOYS"
60 #"": #"ENTER ABOUT 4 LINES OF TEXT"
62 INPUT Q$
64 #"ENTER NEXT 4 LINES OF TEXT."
66 INPUTX$
68 E$=X$+Q$
70 IFLEN(E$)<120 THEN #"SAMPLE SIZE IS TOO SMALL!"
72 IFLEN(E$)<120 THEN 60
74 FORX=1TOLEN(E$)
76 IFE$(X,X)=" " THEN S=S+1 ELSE 88
78 IFX=1 THEN S=S-1
80 IFX=1 THEN 86
82 IFE$(X,X)=E$(X-1,X-1) THEN S=S-1
84 IFE$(X,X)=E$(X-1,X-1) THEN P=P+1
86 NEXT
88 #"":#"THE NUMBER OF WORDS HAVE NOW"
90 #"BEEN COUNTED."
92 IFS<>0 THEN 98
94 #"":#"NO WORDS COUNTED!!"
96 END
98 #"":#"THE NUMBER OF SENTENCES HAVE"
100 #"BEEN DETERMINED."
102 FOR T=1 TO LEN(P$)-2 STEP 3
104 FOR X=3 TO LEN(E$)
106 IFE$(X-2,X)=P$(T,T+2) THEN W=W+1 ELSE 112
108 A=3
110 GOSUB 194
112 NEXT:NEXT
114 FORT=1TOLEN(R$)-1STEP2
116 FOR X=2 TOLEN(E$)
118 IFE$(X-1,X)=R$(T,T+1) THEN W=W+1 ELSE 124
120 A=2
122 GOSUB 194
124 NEXT:NEXT
126 FOR X=1 TO LEN(E$)
128 IFE$(X,X)=I$ THEN W=W+1 ELSE 134
130 A=1
132 GOSUB 194
134 NEXT
136 FORX=1TOLEN(E$)
138 IFE$(X,X)="A" THEN V=V+1
140 IFE$(X,X)="E" THEN V=V+1
142 IFE$(X,X)="I" THEN V=V+1
144 IFE$(X,X)="O" THEN V=V+1
146 IFE$(X,X)="U" THEN V=V+1
148 NEXT
150 #"":#"THE NUMBER OF SYLLABLES"
152 #"HAVE NOW BEEN COUNTED."
154 FOR T=1 TO LEN(D$)-4 STEP 5
156 FOR X=5 TO LEN(E$)
158 IFE$(X-4,X)=D$(T,T+4) THEN W=W+1 ELSE 164
160 A=5
162 GOSUB 194
164 NEXT:NEXT
166 FORT=1TOLEN(H$)-5 STEP 6
168 FORX=6 TO LEN(E$)
170 IFE$(X-5),X)-H$(T,T+5) THEN W=W+1 ELSE 176
172 A=6
174 GOSUB 194
176 NEXT:NEXT
178 FORT=1TOLEN(T$)-3 STEP 4
180 FOR X=4 TOLEN(E$)
182 IFE$(X-3,X)=T$(T,T+3) THEN W=W+1 ELSE 188
184 A=4
186 GOSUB 194
188 NEXT:NEXT
190 GOTO 216
194 IF X=LEN(E$) THEN 210
196 IF E$(X+1,X+1)="." THEN 210
198 IF E$(X+1,X+1)="?" THEN 210
200 IF E$(X+1,X+1)="!" THEN 210
202 IF E$(X+1,X+1)="(" THEN 210
204 IF E$(X+1,X+1)=")" THEN 210
206 IF E$(X+1,X+1)   "  " THEN W=W-1
208 IF E$(X+1,X+1)   "  " THEN 214
210 IF X=A THEN 214
212 IF E$(X-A,X-A)   "  " THEN W=W-1
214 RETURN
216 #"":#"THE NUMBER OF PERSONAL WORDS"
218 #"HAVE BEEN COUNTED."
```

```
220 FORX=1 TO LEN(E$)
222 IFE$(X,X)=“(” THEN Q=Q+1
224 IFE$(X,X)=“!” THEN E=E+1
226 IF E$(X,X)=“?” THEN F=F+1
228 NEXT
230 #“”
232 IF P=O THEN P=.01
234 R=206.835-((S*1.015)/P)-(76.5*V/S)
236 IF R >100 THEN R=100
238 #“READING EASE = ”;R
240 B=(363.5*W/S)+(.314*(Q+E+F))
242 IF B >100 THEN B=100
244 #“HUMAN INTEREST= ”;B
246 #“THE READING EASE SCORE OF”
248 #INT(10*R)/10
250 #“INDICATES THE MATERIAL IS”
252 IF R >90 THEN #“VERY EASY TO READ; 4TH GRADE LEVEL.”
254 IF R >90 THEN 278
256 IF R >80 THEN #“EASY TO READ; 5TH GRADE LEVEL.”
258 IF R >80 THEN 278
260 IF R >70 THEN #“FAIRLY EASY TO READ; 6TH GRADE LEVEL.”
262 IF R >70 THEN 278
264 IF R >60 THEN #“STANDARD DIFFICULTY; 8TH GRADE LEVEL.”
266 IF R >60 THEN 278
268 IF R >50 THEN #“FAIRLY DIFFICULT TO READ; HIGH SCHOOL LEVEL”
270 IF R >50 THEN 278
272 IF R >30 THEN #“ DIFFICULT TO READ; HIGH SCHOOL JR. COLLEGE LEVEL.”
274 IF R >30 THEN 278
276 IF R <30 THEN #“ VERY DIFFICULT TO READ; COLLEGE TEXT LEVEL.”
278 FOR X=1 TO 2000:NEXT
280 #“THE HUMAN INTEREST SCORE OF”
282 #INT(10*B)/10
284 #“INDICATES THAT THE MATERIAL IS”
286 IF B >60 THEN #“DRAMATIC; SIMILAR TO FICTION MATERIAL.”
288 IF B >60 THEN 304
290 IF B >40 THEN #“HIGHLY INTERESTING; SIMILAR TO THAT FOUND IN THE NEW YORKER MAGAZINE.”
292 IF B >40 THEN 304
294 IF B >20 THEN #“INTERESTING; SIMILAR TO MATERIAL FOUND IN DIGESTS.”
296 IF B >20 THEN 304
298 IF B >10 THEN #“MILDLY INTERESTING; SIMILAR TO MATERIAL FOUND IN TRADE JOURNALS.”
300 IF B >10 THEN 304
302 IF B <10 THEN #“DULL; SCIENTIFIC PUBLICATION LEVEL.”
304 END
```

nesting the FOR-NEXT loops for each personal word length into a single routine would decrease memory consumption. This program was run on my Digital Group Z-80 18K system using DGSS Maxi-BASIC Version 1 as the interpreter.

Conclusions

Here we have a program that can be used to objectively characterize the content of written material.

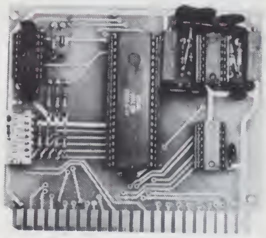
Inclusion of this program in a text editor would yield a software system that could not only format textual material but evaluate it for prescribed content as well.

Next step — give your computer a writing assignment, and stand back! ■

Reference

Rudolf, Flesch, *The Art of Readable Writing*, Harper Bros., New York.

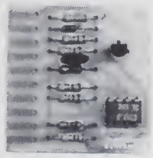
ELECTRONIC SYSTEMS
p.o. box 212 Burlingame CA 94010
(408) 374-5984



UART & BAUD RATE GENERATOR

Part no. 101
- Converts serial to parallel and parallel to serial
- Low cost on board baud rate generator
- Baud rates: 110, 150, 300, 600, 1200, and 2400
- Low power drain +5 volts and -12 volts required
- TTL compatible
- All characters contain a start bit, 5 to 8 data bits, 1 or 2 stop bits, and either odd or even parity.
- All connections go to a 44 pin gold plated edge connector
- Board only $12.00; with parts $35.00

RS-232/TTL INTERFACE



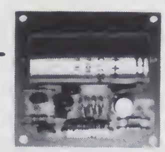
Part no. 232
- Converts TTL to RS-232, and converts RS-232 to TTL
- Two separate circuits
- Requires -12 and +12 volts
- All connections go to a 10 pin gold plated edge connector
- Board only $4.50; with parts $7.00

DC POWER SUPPLY

Part no. 6085
- Board supplies a regulated +5 volts at 3 amps., +12,-12, and -5 volts at 1 amp.
- Board has filters, rectifiers, and regulators
- Power required is 8 volts AC at 3 amps., and 24 volts AC C.T. at 1.5 amps.
- Board only $12.50

TAPE INTERFACE

Part no. 111
- Play and record Kansas City Standard tapes
- Converts a low cost tape recorder to a digital recorder
- Works up to 1200 baud
- Digital in and out are TTL-serial
- Output of board connects to mic. in of recorder
- Earphone of recorder connects to input on board
- Requires +5 volts, low power drain
- Board $7.60; with parts $27.50
- No coils

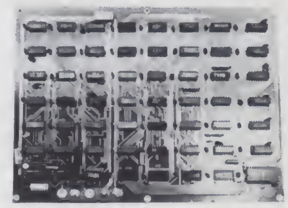


Part no. 107

RF MODULATOR

- Converts video to AM modulated RF, Channels 2 or 3
- Power required is 12 volts AC C.T., or +5 volts DC
- Board $4.50; with parts $13.50

TELEVISION TYPEWRITER



Part no. 106
- Stand alone TVT
- 32 char/line, 16 lines, modifications for 64 char/line included
- Parallel ASCII (TTL) input
- Video output
- 1K on board memory
- Output for computer controlled curser
- Auto scroll
- Non-distructive curser
- Curser inputs: up, down, left, right, home, EOL, EOS
- Scroll up, down
- Requires +5 volts at 1.5 amps, and -12 volts at 30 mA
- Board only $39.00; with parts $145.00

8K STATIC RAM

Part no. 300
- 8K Altair bus memory
- Uses 2102 Static memory chips
- Fully by-pass memory protect
- Gold contacts
- Wait states
- On board regulator
- S-100 bus compatible
- Vector input option
- TRI state buffered
- Board only $22.50; with parts $160.00

TIDMA

Part no. 112
- Tape Interface Direct Memory Access
- Record and play programs without bootstrap loader (no prom) has FSK encoder/decoder for direct connections to low cost recorder at 625 baud rate, and direct connections for inputs and outputs to a digital recorder at any baud rate.
- S-100 bus compatible
- Comes assembled and tested for $160.00

APPLE I MOTHER BOARD

Part no. 102
- 10 slots — 44 pin (.156) connectors spaced ¾" apart
- Connects to edge connector of computer
- Pin 20 and 22 connects to X & Z for power and ground
- Board has provisions for by-pass capacitors
- Board cost $15.00

MODEM

Part no. 109
- Type 103
- Full or half duplex
- Works up to 300 baud
- Originate or Answer
- No coils, only low cost components
- TTL input and output-serial
- Connect 8 ohm speaker and crystal mic. directly to board
- Uses XR FSK demodulator
- Requires +5 volts
- Board $7.60; with parts $27.50

To Order:

Mention part number and description. For parts kits add "A" to part number. Shipping paid for orders accompanied by check, money order, or Master Charge, BankAmericard, or VISA number, expiration date and signature. Shipping charges added to C.O.D. orders. California residents add 6.5% for tax. Parts kits include sockets for all ICs, components, and circuit board. Documentation is included with all products. Dealer inquiries invited. 24 Hour Order Line: (408) 374-5984.

E21

All in One

George Morrow's versatile front-panel board

George Baeslack

At the first sign of an "affordable" computer, namely the Altair 8800, I wanted one, but I didn't have enough money to buy the machine and the necessary cards. Later, in the bicentennial year, I finally accumulated enough to buy the basic machine. Unfortunately, the price of kits using the 8080a CPU had also increased, and again I was unable to afford the whole kit. I then decided to look for lower-priced alternatives to the Altair and Imsai.

My Pre-Morrow Efforts

Buying the individual pieces and wire-wrapping them seemed to be the best approach. So, I sent an order to the Southern California Computer Society's (S.C.C.S.) group-purchase organization, for the CPU chip, clock chip, 8212 chip, a 20 Amp transformer, Altair bus 15-slot motherboard, edge connectors and lots of bus drivers.

While waiting for the parts, I started to build the cabinet and chassis. I made the box from 3/32 inch sheet steel using a box break (a table used to bend sheet metal at right angles). The cabinet measures 19 X 17 X 7 inches, and is finished in wood-grain vinyl, which was included with the metal.

As the parts trickled in, I started on other phases of the project. The first piece to arrive was the 20 Amp transformer. (Group purchase is slow because a certain number of orders are required before a particular item is shipped.)

The transformer produced 16 volts at 20 Amps center-tapped, and 32 volts at 2 Amps center-tapped. The proper voltages can be obtained using a simple rectifying circuit. One or more diodes may have to be added serially if the power-supply voltage becomes too high due to current underutilization and lack of alternate primary power taps. The electrolytic capacitors were bought at a local surplus shop; wire, terminal strips, fuses and other hardware came from the local Radio Shack.

The second package from S.C.C.S. contained the edge connectors, 8212 8-bit latches, and clock driver. Now I could wire-wrap the IC sockets for a CPU board. When you first look at a PC card (e.g., an Altair CPU card) it looks simple. When you start to wire-wrap the same design, it becomes long, hard work, with wires going everywhere—sometimes to the wrong places. After wiring the prototype board, I installed the few components I had and set it aside.

While I was waiting patiently for the rest of the parts from group purchase, I noticed that Morrow's Micro-Stuff was advertising an 8080 front panel/CPU board that would have nicely fit my budget. It featured an 8080a CPU and an S-100-compatible board that would easily fit in the box I had already made. It had a smart front panel to talk to memory and registers at the push of a button. Very nice indeed. Oh, well . . . if I had only waited a little longer.

After a long time, and many unanwered requests for information about the status of my parts, I was finally notified that group purchase had gone bust and my order would not be completed. Phooey! Now, with a box and a power supply, I was stuck with ordering the rest of the parts from other distributors. But—now I was not committed to my original design! I thought it over carefully and decided to go with Morrow's front panel/CPU card. This would free me from all the messy wiring that goes with hooking the front panel to the bus.

Building the Morrow Front Panel Board

About two weeks after I sent my check, I received a package from Morrow's Micro-Stuff containing a sophisticated PC board, schematics, black-and-white glossy of the finished board, an assembly-instruction manual, Intel's 8080 data book and Osborne's *Programming for Logic Design*.

The kit-building section gives a few tips before you begin soldering: Place the components as they are called for, using the parts-placement diagram. The diagram was easy to follow, and I finished the board in one night, soldering in all IC sockets and other components. (Note: There was no warning notice that the 8080a chip is static sensitive. I knew about this sensitivity, but a novice might not, and could easily zap this chip before installing it!) After I inserted all ICs I connected the power supply, and it worked! The digits lighted, and all seemed to work perfectly.

Data Entry and Other Thoughts

First, data can be entered into the hardware (memory, CPU registers and I/O ports) via push buttons. The radix used for entering data is octal; hexadecimal numbers cannot be entered directly into the machine and must be translated into equivalent values.

You say you don't like octal, particularly when most software is created in hex? Wait a minute. Remember those LEDs on the Imsai and Altair front panels? Well, they have been replaced by ten seven-segment LEDs. One full octal number can be represented by each seven-segment LED, instead of the usual three single LEDs that represent each octal constant. Decoding addresses is accomplished by simply reading the values on the displays, instead of the bit-by-bit decoding that must be done on the Altair and Imsai machines.

Now, what about entering data into memory? Assuming the address has been set on the Imsai, you must use the eight data switches to enter a value. This can become quite

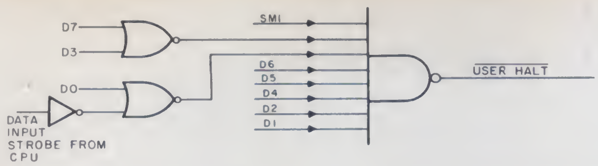


Fig. 1. The basic circuitry to decode the halt condition when SM1 (start of instruction fetch cycle), the halt instruction and a CPU data input a request. This hardware can be incorporated in an Imsai 8080 to provide the user with "controlled halt."

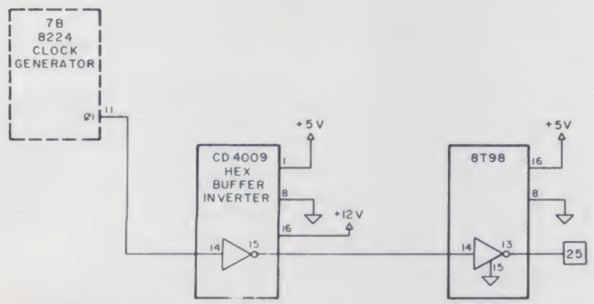


Fig. 2. A patch for the Morrow board to get Ø1 on the bus.

tedious, especially if you have to enter an entire program (like Intel's paper-tape loader). With the Morrow board, only three buttons have to be pressed. Each button, numbered 0-7, represents three binary bits of data. It is much easier to enter octal instead of binary digits!

The rest of the front-panel switches, Deposit (byte in memory), Deposit Next (byte in memory), Examine (location in memory) and Examine Next (location in memory) provide normal front-panel functions.

The Altair's single-step function lets you go through each cycle in the execution of an instruction, one for each depression of the switch, or a few at a time (slow-stepping) for the duration of the key depression. This works fine, but what happens if you want to go a little faster (or slower) in the slow-stepping mode? To change the slow-stepping speed, you must go back to the hardware and change a few jumpers. On the Morrow board, you can single-step or slow-step through your program at any rate you wish, simply by depressing the desired delay on the front-panel keyboard. (Note: Only the instruction that is to be executed will show up on the front-panel display.)

Now that I've tried to compare a few features of the Altair 8800, 8800a and Imsai 8080, let's look at the advanced Altair 8800b. Altair incorporated features in this model to let the user examine and deposit data into the accumulator. You can also examine and deposit data to I/O device ports using the accumulator as data and address lights showing to which port the data should go. To perform these functions, Altair has a PROM program that runs the front panel.

Morrow's board can perform both of these functions. Mode 1 lets you deposit and examine the accumulator and any other available register that is on the 8080 CPU chip. To get data to and from I/O ports, simply get into mode 2, select the port you want, and examine or deposit to any port. This is done without altering the contents of the accumulator.

So what's different? If you've ever used an Imsai or Altair, you will notice that the halt instruction doesn't halt as most computers do—the CPU goes into an internal loop and stays there. Morrow's board detects the halt instruction before the CPU can get at it and activates the front-panel program instead. Mr. Morrow calls it a "controlled halt" condition. You can set breakpoints with the HLT code, examine the progress of your program and continue on till the next breakpoint.

Remember that slow-step (not single-step) feature? It works just as well in the register and I/O port modes. So, while your program is slow-stepping away, you can observe the internal happenings of any register and see exactly what happens in each port. To the inexperienced programmer, slow stepping, and observing what goes on internally, is a great advantage. It lets you see your mistakes as they happen, but before they make gibberish of everything else that follows.

Now that you see the advantages of the front panel/CPU board, you might ask, "What's the catch?" The catch is that all these features are run by a program incorporated in the front-panel circuitry, which uses 512 bytes of memory. The front-panel program is located in the last two pages (page = 256 bytes) of available memory and is protected from user meddling. However, the LED display is not protected, so the user can display data on it when the front panel doesn't have control. This is done simply by writing out to certain memory locations, which happen to be the addresses of the LEDs.

Operating the Front Panel

Using the front panel is the best part of the project. I borrowed a 4K board and started to learn the four modes of the panel. The ten seven-segment LEDs are used to display all information in octal and can be used to display data from a program not controlled by the front panel.

When you turn on the system, it is always in mode 0; this allows you to examine and fill the memory. For example, examine location 343, press 3, 4, 3 and the E key; you are now observing the contents of location 343 (the left part of display shows the address, and the right half shows the contents of that address). To examine the next location, simply press E again.

To deposit a byte, you must press E (examine that location first), type in the byte that is to replace the current contents and press D (deposit). To deposit next, enter another byte and press D again. The address is incremented by one, and the byte keyed in is now in that memory location.

Mode 1 allows you to examine and fill any CPU

Front Panel Board—The cost for this item is about $250 from Morrow's Micro-Stuff. I have not been able to find another set of boards that would fit my needs and budget.

S-100 Motherboard—The 15-slot motherboard is made by Solid State Music and sold by Mikos for $45. Of the 15 slots, 13 are usable for peripheral cards (one slot was used for the panel board, the other was covered up by the support backplane for the front panel). A variety of motherboards is now available in different sizes and card capacities, and with different features.

Power Supply—This item was made from a transformer bought from group purchase for $20. The capacitors, diodes and heat sinks cost about $20 at a local surplus shop. An article describing these transformer and power-supply techniques appeared in *Kilobaud* No. 8, p. 50.

Hardware—The sheet metal and C bars cost about $25. I am assuming that screws, rubber feet and lots of other parts are in your junk box—otherwise, add another $10-$20 to the cost.

Labor—An ingredient used extensively throughout this project. The value is self-determined, and therefore cannot be estimated by the author. I used about 1-1½ weeknights (about 6-11 pm) in making the cabinets and fitting all the parts together. If you are not handy at this type of work, I would not recommend the project.

Table 1. Parts list.

register, including the Program Counter, Stack Pointer and the Program Status registers. You can change any individual register, or pair of registers, by entering the register address (described in the front-panel instruction manual) and pressing E. To examine the next register, press E again. To change the contents, enter the data after examining that register and press D. It's that simple.

Mode 2 allows you to examine and deposit bytes into I/O ports—a neat feature when debugging a port. Enter the port number, press E, and you have examined the byte made available by the port. Examine Next and Deposit Next do not exist for this mode, and subsequent Es and Ds will examine and deposit to that port.

Mode 3 is like mode 0, except when slow-stepping. The S key is used to slow-step and initiate slow-stepping. A single press of the S executes one instruction pointed to by the PC register (*not* by the address on the front-panel LEDs). By pressing a number and S, you will initiate the slow-stepping feature. Your program will start executing at the rate of the number keyed in multiplied by the delay factor.

Slow-stepping in each mode produces different results. In mode 0 it displays the current address and the first byte of each instruction. It is helpful to see program control in action. In mode 1 it displays the contents of the last register examined. As the program executes, any changes made to the register being displayed will be updated by the front panel. Mode 2 will let you observe the transfer of bytes to the I/O ports while the program is running, and mode 3 will indicate any changes that have been made to the last location observed.

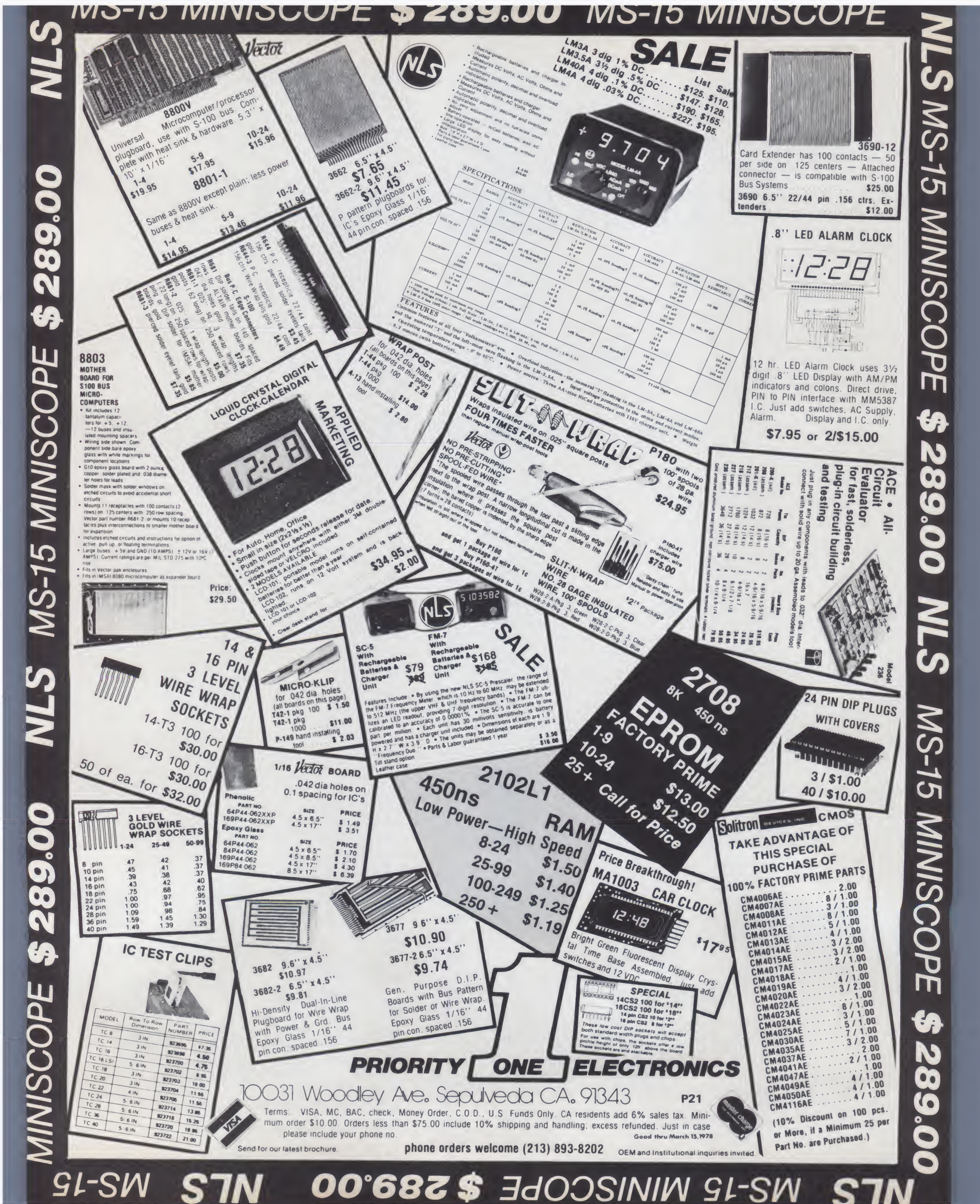
Finally, to run your program at full speed, press only the M key. The display becomes inert unless your program wants to display data on the LEDs. The program is now running without front-panel control. To stop it, press S, and the front panel comes alive again.

The completed system.

Closing Notes

The Morrow front panel/CPU board is an ingenious piece of electronic workmanship, but it has flaws—one being that there are only six support holes for 16 inches of board!

After a lot of work assembling the power supply and motherboard, and designing and fabricating the cabinet, I feel it was worth it. Those who are handy and hardy enough will get a lot of enjoyment from creating a computer they can call their own.■



NLS MS-15 MINISCOPE $289.00
8800V
Universal Microcomputer/processor plugboard, use with S-100 bus. Complete with heat sink & hardware. 5.3'' x 10'' x 1/16''
1-4 $19.95
5-9 $17.95
10-24 $15.96
8801-1
Same as 8800V except plain; less power buses & heat sink.
1-4 $14.95
5-9 $13.46
10-24 $11.96
3662 6.5'' x 4.5'' $7.65
3662-2 9.6'' x 4.5'' $11.45
P pattern plugboards for IC's Epoxy Glass 1/16'' 44 pin con. spaced .156
SALE
LM3A 3 dig 1% DC .... $125. $110.
LM3.5A 3½ dig .5% DC ...... $147. $128.
LM40A 4 dig .1% DC ...... $190. $165.
LM4A 4 dig .03% DC ...... $227. $195.
SPECIFICATIONS
FEATURES
3690-12
Card Extender has 100 contacts — 50 per side on .125 centers — Attached connector — is compatible with S-100 Bus Systems $25.00
3690 6.5'' 22/44 pin .156 ctrs. Extenders $12.00
.8'' LED ALARM CLOCK
12 hr. LED Alarm Clock uses 3½ digit .8'' LED Display with AM/PM indicators and colons. Direct drive, PIN to PIN interface with MM5387 I.C. Just add switches, AC Supply, Alarm. Display and I.C. only.
$7.95 or 2/$15.00
8803
MOTHER BOARD FOR S100 BUS MICRO-COMPUTERS
Price: $29.50
LIQUID CRYSTAL DIGITAL CLOCK-CALENDAR
APPLIED MARKETING
$34.95
$2.00
WRAP POST
SLIT-N-WRAP
FOUR TIMES FASTER
P180
$24.95
$75.00
ACE • All-Circuit Evaluator for fast, solderless, plug-in circuit building and testing
14 & 16 PIN 3 LEVEL WIRE WRAP SOCKETS
14-T3 100 for $30.00
16-T3 100 for $30.00
50 of ea. for $32.00
MICRO-KLIP
for .042 dia. holes (all boards on this page)
T42-1 pkg 100 $1.50
T42-1 pkg 1000 $11.00
P-149 hand installing tool $2.03
SC-5 With Rechargeable Batteries & Charger Unit $79
FM-7 With Rechargeable Batteries & Charger Unit $168
SALE
2708 8K 450 ns EPROM FACTORY PRIME
1-9 $13.00
10-24 $12.50
25+ Call for Price
24 PIN DIP PLUGS WITH COVERS
3 / $1.00
40 / $10.00
1/16 Vector BOARD
.042 dia holes on 0.1 spacing for IC's
2102L1 450ns RAM Low Power—High Speed
8-24 $1.50
25-99 $1.40
100-249 $1.25
250+ $1.19
3 LEVEL GOLD WIRE WRAP SOCKETS
Price Breakthrough! MA1003 CAR CLOCK
Bright Green Fluorescent Display Crystal Time Base Assembled. Just add switches and 12 VDC.
$17.95
Solitron CMOS
TAKE ADVANTAGE OF THIS SPECIAL PURCHASE OF 100% FACTORY PRIME PARTS
IC TEST CLIPS
3682 9.6'' x 4.5'' $10.97
3682-2 6.5'' x 4.5'' $9.81
Hi-Density Dual-In-Line Plugboard for Wire Wrap with Power & Grd. Bus Epoxy Glass 1/16'' 44 pin con. spaced .156
3677 9.6'' x 4.5'' $10.90
3677-2 6.5'' x 4.5'' $9.74
Gen. Purpose D.I.P. Boards with Bus Pattern for Solder or Wire Wrap. Epoxy Glass 1/16'' 44 pin con. spaced .156
SPECIAL
(10% Discount on 100 pcs. or More, if a Minimum 25 per Part No. are Purchased.)
PRIORITY ONE ELECTRONICS
10031 Woodley Ave. Sepulveda CA. 91343
P21
Terms: VISA, MC, BAC, check, Money Order, C.O.D., U.S. Funds Only. CA residents add 6% sales tax. Minimum order $10.00. Orders less than $75.00 include 10% shipping and handling; excess refunded. Just in case please include your phone no.
Good thru March 15, 1978
Send for our latest brochure.
phone orders welcome (213) 893-8202
OEM and Institutional inquiries invited.

Deflection!

a video game for the quick and agile

Andrew A. Recupero
1388 Kinsport Lane
San Jose CA 95120

Here's a fast-moving, not-so-dull video game for owners of Processor Technology's VDM-1, Polymorphic's VTI 64, Solid State Music's VB1, the Cromemco Dazzler and equivalent video display systems. We would all like to see more games like this. — John.

After becoming hooked on those arcade-type video games, my prime ambition in personal computing has been to duplicate and create my own video games on my home computer. This ambition led to programming this game I call Deflection.

The game uses a video display and a keyboard and consists of a playing field, a runner and one or three targets. The playing field is the video display. The runner is constantly in motion on the field at a relatively high rate of speed. He reverses direction whenever he reaches the edge of the playing field.

How To Play

The object of the game is for you, the player, to deflect the runner so that he strikes a target. A timer limits the playing time and to win you must hit all the targets within the preset time.

You deflect the runner by inserting a deflector in his path. This is done by carefully timing an input from the keyboard. The deflector causes a 90 degree change in the runner's direction. Once a deflector has been added to the field, it remains for the entire game, acting as both an aid and an obstacle to future plays. You continue to add deflectors until either time expires or all the targets are hit.

The keyboard inputs for inserting deflectors are either of the two ASCII slash keys / or \. Which key you press depends on the runner's direction and where you want him to go next. For example, if the runner is moving left to right and you want him to go up, you would type in a /. If the runner is moving from right to left, this same input would cause the runner to go down.

The game has two versions to make it slightly more difficult. The first is to start the game with three targets instead of one. The second version omits clearing the field of the deflectors from the previous game. In this version the deflectors are shifted one position to the left to avoid exactly duplicating the previous game. The second version is only possible if you win the previous game since the field is automatically cleared after the timer expires.

To start each game you type in a number 0, 1, 2, or 3. These are interpreted as follows:

Input 0 = 1 target, clear field.

Input 1 = 1 target, shift deflectors.

Input 2 = 3 targets, clear field.

Input 3 = 3 targets, shift deflectors.

Installing the Program

I tried to simplify the process of customizing this program for your own system by highlighting at the right edge of the program listing (--X) those instructions which you might have to change. There are three types noted. Type 1 (--1) are the various

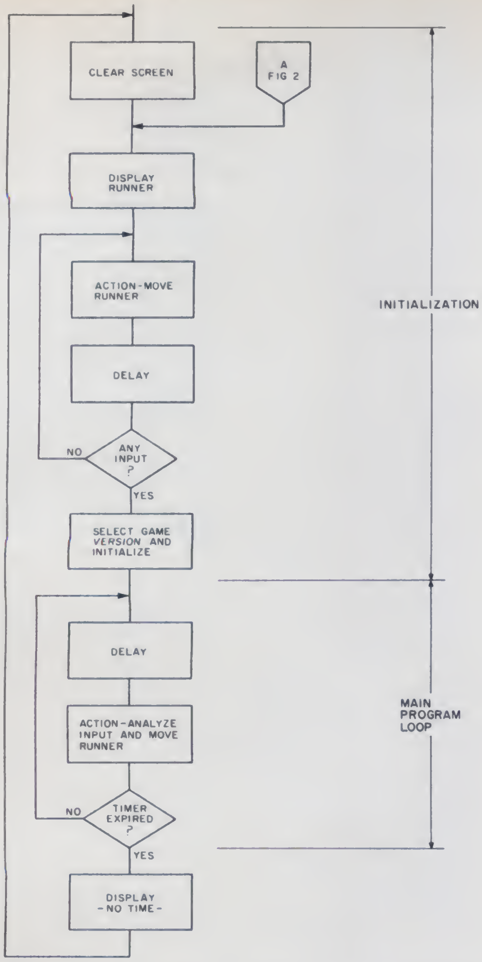


Fig. 1. Flowchart for the main part of the program.

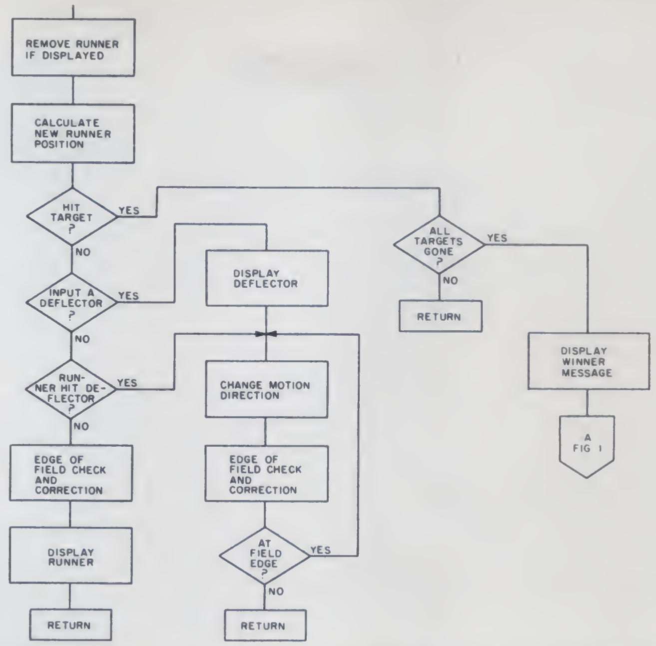


Fig. 2. Flowchart for the subroutine called Action. This subroutine contains all of the game logic.

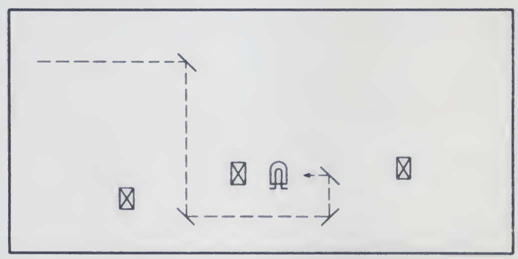


Fig. 3. Sketch of typical mid-game display showing runner that has been deflected a number of times heading toward one of the three targets.

timer initialization instructions. Type 2 (--2) are the displayed characters for the runner, targets and deflectors. These instructions can be changed to suit the character ROM used in your display. Type 3 (--3) are the keyboard ports and video buffer addresses. The program was assembled for a VDM-1 with the video buffer at hex CC00. The listing has all the video buffer addresses referenced from the base video buffer address to simplify conversion.

This game must be run on a video display that allows direct program access to the display buffer. This type of display usually treats the display buffer as part of memory and is currently available from a number of manufacturers. The TVT type display with a serial interface will not work since the display update ability from the computer is too slow.

Program Description

The program as shown was written for an 8080 with a video display having 16 lines with 64 characters per line. The program can be adapted to other microprocessors and display formats.

The main program flowchart is shown in Fig. 1. There are two main sectlons to this part of the program. The first is concerned with initialization. This part keeps some action going on the display while waiting for the game option to be keyed in. After the game option is selected, the playing field is initialized for the start of play.

The second part handles all the game playing and controls the game timer. Here the program is in the form of a loop which calls two subroutines and updates the game timer on each pass through the loop. The two subroutines used are called Delay and Action.

The subroutine called Delay controls the time period between successive moves of the runner. During this delay period the keyboard is checked for player input. Any player input is saved and passed to the Action subroutine in the C register.

The heart of all the game logic is in the subroutine called Action. The flowchart for this subroutine is shown in Fig. 2. This routine calculates the runner's new location and then checks to see if he has hit a target or a deflector. If a target is hit, the target count is decremented. The player wins if this count is zero.

If a deflector is hit, the runner's direction is changed. This depends on the current runner direction and the type of deflector hit. If the player's input, if any, is valid, it is displayed and treated as an automatic deflector hit. After hitting a deflector, a check is made for the edge of the playing field, in which case the runner's motion flag is reversed. If he is at the field edge, a second change of direction must be made since the runner has automatically hit the same deflector from a different direction. The runner is not dispiayed after hitting a deflector to *preserve* the deflectors on the playing field.

If neither a target nor a deflector was hit, a check for the edge of the playing field is made and the runner is displayed at the new location.

The comments in the program listing should provide any additional details, if needed.

Final Notes

One final note on playing is in order. Since the runner is not displayed when he hits a deflector, it is possible to completely box in the runner with deflectors so as to make him invisible. However, he is still there, doing his thing, and the game timer is still running! The way out of this is to enter another deflector so as to un-box the runner.

The difficulty of the game can be increased or decreased by merely changing the game duration timer or the delay timer. For some idle amusement, the game timer can be disabled and the runner allowed to do his thing unending.

I hope this program will encourage others to program and share action games for the video display. If you are looking for some game ideas, I suggest some discussions with friends with similar interests. The basic idea for this game was given to me by a friend. A visit to a video game arcade is also helpful. The professional video games offer a number of ideas which could be adapted or used to stimulate original games for the home computer. Have fun! ■

Program.

```
                 *  THE GAME OF DEFLECTION        BY ANDREW RECUPERO
                 *     EXECUTE PROGRAM AT ADDRESS HEX 0050
                 *     TO START TYPE IN 0,1,2 OR 3, INTERPRETED AS FOLLOWS:
                 *                               INPUT 0 = 1 TARGET, CLEAR FIELD
                 *                                     1 = 1 TARGET SHIFT DEFLECTORS
                 *                                     2 = 3 TARGETS, CLEAR FIELD
                 *                                     3 = 3 TARGETS SHIFT DEFLECTORS
                 *     OBJECT IS TO HIT TARGET(S) BY DEFLECTING RUNNER TO TARGET.
                 *     DEFLECT RUNNER BY TYPING IN A '/' OR A '\'. ALL TARGETS MUST
                 *     BE HIT WITHIN A FIXED TIME PERIOD.
                 *
                 *     NUMBERS AT RIGHT BOARDER (--X) INDICATE STATEMENTS WHICH MAY
                 *     REQUIRE PERSONALIZATION
                 *
                 STATUS    EQU   X'06'            KEYBOARD STATUS PORT
                 STROBE    EQU   X'02'            KEYBOARD STROBE BIT
                 DATA      EQU   X'05'            KEYBOARD DATA PORT
                 VDMRAM    EQU   X'CC00'          VDM RAM STARTING ADDRESS
                 VDMPAGE   EQU   X'CC'            VDM RAM PAGE NUMBER
                 VDMPORT   EQU   X'C8'            VDM PORT ADDRESS
                 TARGET    EQU   X'05'            TARGET SYMBOL
                 RUNNER    EQU   X'07'            RUNNER SYMBOL
                 USLSH     EQU   X'2F'            UP SLASH CODE '/'
                 DSLSH     EQU   X'5C'            DOWN SLASH CODE '\'
                 LEFT      EQU   X'80'            RUNNER
                 DOWN      EQU   X'40'                DIRECTION
                 UP        EQU   X'20'                    CONTROL
                 RIGHT     EQU   X'10'                        EQUATES
                 *
0050   31EC01    GSTART    LXI   SP,STACK+8       INITIALIZE STACK POINTER
0053   CD9F01              CALL  CLSCRN           CLEAR SCREEN
0056   CDC200    REIDLE    CALL  DRUNER           DISPLAY RUNNER
0059   CDCD00    IDLE      CALL  ACTION           MOVE RUNNER
005C   CD8501              CALL  DELAY            EXECUTE TIME DELAY AND CHECK INPUT
005F   79                  MOV   A,C              KEYBOARD INPUT TO ACCUMULATOR
0060   B7                  ORA   A                TEST FOR NON-ZERO
0061   CA5900              JZ    IDLE
0064   3620                MVI   M,X'20'          REMOVE RUNNER
0066   0F                  RRC                    SHIFT DEFLECTORS?
0067   F5                  PUSH  PSW              SAVE GAME OPTION IN A REG
0068   DA7100              JC    SHIFT            JMP TO SHIFT IF YES
006B   CD9F01              CALL  CLSCRN           DO NOT SHIFT, CLEAR SCREEN
006E   C38100              JMP   INIT
0071   0100CC    SHIFT     LXI   BC,VDMRAM          SHIFT                    --3
0074   50                  MOV   D,B                  SCREEN
0075   59                  MOV   E,C                     ONE
0076   13                  INX   DE                        POSITION
0077   1A        SMOR      LDAX  DE                           TO
0078   02                  STAX  BC                             THE
0079   03                  INX   BC                                LEFT
007A   13                  INX   DE
007B   7A                  MOV   A,D
007C   FED0                CPI   VDMPAGE+4                                   --3
007E   C27700              JNZ   SMOR             GO SHIFT SOME MORE IF NOT DONE
0081   F1        INIT      POP   PSW              RESTORE GAME OPTION TO A REG
0082   0E01                MVI   C,1              SET FOR ONE TARGET
0084   0F                  RRC                    GAME OPTION BIT 0 TO CARRY FLAG
0085   3E05                MVI   A,TARGET         TARGET SYMBOL TO A REG      --2
0087   D29200              JNC   GAM1             JMP FOR 1 TARGET OPTION
008A   0E03                MVI   C,3              INITIALIZE FOR 3 TARGETS
008C   324CCF              STA   VDMRAM+X'034C'PUT TARGET TO LEFT OF CENTER  --3
008F   32F4CE              STA   VDMRAM+X'02F4'TARGET TO THE RIGHT           --3
0092   3220CF    GAM1      STA   VDMRAM+X'0320'TARGET IN THE MIDDLE          --3
0095   79                  MOV   A,C
0096   32E301              STA   TARNUM           STORE NUMBER OF TARGETS
0099   21C001              LXI   HL,X'01C0'       INITIALIZE GAME TIMER         --1
009C   22E101              SHLD  TIMER            SAVE TIMER VALUE
009F   CDC200              CALL  DRUNER           START GAME — DISPLAY RUNNER
                 * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
                 *  THIS IS THE MAIN LOOP FOR THE GAME
                 *
00A2   CD8501    MAINLP    CALL  DELAY            WAIT AWHILE & CHECK KEYBOARD
```

```
00A5   CDCD00                  CALL   ACTION
00A8   E5                      PUSH   HL                  SAVE HL
00A9   2AE101                  LHLD   TIMER               GET GAME TIMER
00AC   2B                      DCX    HL                  DECREMENT TIMER
00AD   7C                      MOV    A,H                 CHECK IF GAME
00AE   B5                      ORA    L                        TIMER EXPIRED
00AF   22E101                  SHLD   TIMER               STORE UPDATED TIMER
00B2   E1                      POP    HL                  RESTORE HL
00B3   C2A200                  JNZ    MAINLP              JMP IF TIMER DID NOT EXPIRE
00B6   01D301                  LXI    BC,LOSE             DISPLAY NO
00B9   CDAF01                  CALL   SHOW                     TIME MESSAGE
00BC   CDBD01                  CALL   WAER                WAIT AWHILE THEN ERRASE MESSAGE
00BF   C35000                  JMP    GSTART              TIMER EXPIRED — RESTART GAME
                 ***********************************************************
                 *  THIS ROUTINE INITIALIZES AND DISPLAYS RUNNER
                 *                       B REG HOLDS MOTION DIRECTION — CODED AS
                 *                         REG BIT 7 6 5 4  3 2 1 0
                 *                                 ! ! ! !  - - - -
                 *                                 ! ! ! RIGHT
                 *                                 ! ! UP
                 *                                 ! DOWN
                 *                                 LEFT
                 *
00C2   2140CC    DRUNER        LXI    HL,VDMRAM+X'0040'   RUNNER STARTING POSITION     --3
00C5   110001                  LXI    DE,X'0100'          X,Y RUNNER POSITION, D=VERT, E=HORZ
00C8   3607                    MVI    M,RUNNER            DISPLAY RUNNER               --2
00CA   0610                    MVI    B,RIGHT             INITIALIZE MOTION TO RIGHT
00CC   C9                      RET
                 ***********************************************************
                 *  THIS ROUTINE DOES ALL THE RUNNER MOVEMENT
                 *
00CD   7E        ACTION        MOV    A,M                 CHECK IF
00CE   FE07                    CPI    RUNNER                   RUNNER                  --2
00D0   C2D500                  JNZ    DF010                        DISPLAYED
00D3   3620                    MVI    M,X'20'             REMOVE RUNNER FROM FIELD
                 *
                 *       CALCULATE NEW RUNNER POSITION
                 *
00D5   78        DF010         MOV    A,B
00D6   FE10                    CPI    RIGHT               IS MOTION TO RIGHT?
00D8   C2DC00                  JNZ    DF012               JMP IF NO
00DB   1C                      INR    E                   MOVE ONE POSITION TO RIGHT
00DC   FE20      DF012         CPI    UP                  IS MOTION UP?
00DE   C2E200                  JNZ    DF014               JMP IF NO
00E1   15                      DCR    D                   MOVE ONE POSITION UP
00E2   FE40      DF014         CPI    DOWN                IS MOTION DOWN?
00E4   C2E800                  JNZ    DF016               JMP IF NO
00E7   14                      INR    D                   MOVE ONE POSITION DOWN
00E8   FE80      DF016         CPI    LEFT                IS MOTION TO LEFT?
00EA   C2EE00                  JNZ    DF018               JMP IF NO
00ED   1D                      DCR    E                   MOVE ONE POSITION TO LEFT
                 *
                 *  CALCULATE FIELD POSITION FROM HORIZONTAL AND VERTICAL VALUES
                 *                                IN D AND E REG, RESULT IN H AND L
00EE   7A        DF018         MOV    A,D
00EF   0F                      RRC
00F0   0F                      RRC
00F1   6F                      MOV    L,A
00F2   E603                    ANI    X'03'
00F4   F6CC                    ORI    VDMPAGE                                          --3
00F6   67                      MOV    H,A                 SAVE HIGH PART OF ADDRESS
00F7   7D                      MOV    A,L
00F8   E6C0                    ANI    X'C0'
00FA   83                      ADD    E
00FB   6F                      MOV    L,A                 SAVE LOW PART OF ADDRESS
                 *
00FC   7E                      MOV    A,M                 CHECK NEXT FIELD POSITION
00FD   FE05                    CPI    TARGET              IS IT A TARGET?              --2
00FF   C21901                  JNZ    DF019               JMP IF DID NOT HIT A TARGET
                 *
                 *  HIT TARGET
                 *
0102   3620                    MVI    M,X'20'             REMOVE TARGET
0104   3AE301                  LDA    TARNUM              GET NUMBER OF TARGETS LEFT
0107   3D                      DCR    A                   REDUCE NUMBER BY ONE
0108   32E301                  STA    TARNUM              STORE UPDATED TARGET COUNT
010B   C0                      RNZ                        RETURN IF MORE TARGETS
010C   D1                      POP    DE                  ALL TARGETS GONE — FIX STACK POINTER
010D   01CC01                  LXI    BC,WIN              DISPLAY
0110   CDAF01                  CALL   SHOW                     WIN MESSAGE
0113   CDBD01                  CALL   WAER                WAIT AWHILE THEN ERASE MESSAGE
0116   C35600                  JMP    REIDLE              GO TO IDLE LOOP
                 *
0119   79        DF019         MOV    A,C                 PUT KEYBOARD DATA INTO A REG
011A   FE2F                    CPI    USLSH               IS IT AN UP SLASH? '/'       --2
011C   CA3001                  JZ     DF020               JMP IF YES
011F   FE5C                    CPI    DSLSH               IS IT A DOWN SLASH? '\'      --2
0121   CA3001                  JZ     DF020               JMP TO ANALYZE DEFLECTOR IF INPUT
                 *                                                                  VALID
                 *         CHECK FOR RUNNER HITTING A DEFLECTOR
```

```
                    *
0124  7E        DFLCK    MOV   A,M
0125  FE20               CPI   X'20'         IS NEXT POSITION BLANK?
0127  C23101             JNZ   CHGMOT        DEFLECTOR IF NO — CHANGE DIRECTION
012A  CD5B01             CALL  EDGCK         REVERSE MOTION IF AT EDGE OF FIELD
012D  3607               MVI   M,RUNNER      DISPLAY RUNNER AT NEW LOCATION     --2
012F  C9                 RET
0130  77        DF020    MOV   M,A           INPUT WAS VALID, DISPLAY IT
                *
                *  CHANGE DIRECTION OF RUNNER MOTION
                *
0131  FE5C      CHGMOT   CPI   DSLSH         IS IT A DOWN SLASH?                --2
0133  78                 MOV   A,B           DIRECTION FLAG TO A REG
0134  C24701             JNZ   UPSLH         JMP IF NOT DOWN SLASH
0137  E6C0               ANI   X'C0'         IS MOTION LEFT OR DOWN
0139  78                 MOV   A,B           RESTORE DIRECTION  FLAG TO A REG
013A  CA4201             JZ    DF030         JMP IF NOT LEFT OR DOWN
013D  0F                 RRC                 CHANGE DIRECTION TO
013E  0F                 RRC                 RIGHT OR UP
013F  C35201             JMP   DF050
0142  07        DF030    RLC                 CHANGE DIRECTION TO
0143  07                 RLC                        LEFT OR DOWN
0144  C35201             JMP   DF050
0147  E6A0      UPSLH    ANI   X'A0'         IS MOTION TO LEFT OR UP?
0149  78                 MOV   A,B           DIRECTION FLAG TO A REG
014A  CA5101             JZ    DF040         JMP IF NOT LEFT OR UP
014D  0F                 RRC                 CHANGE DIRECTION TO RIGHT OR DOWN
014E  C35201             JMP   DF050
0151  07        DF040    RLC                 CHANGE MOTION TO RIGHT OR DOWN
0152  47        DF050    MOV   B,A           SAVE NEW DIRECTION
0153  CD5B01             CALL  EDGCK         IS RUNNER AT EDGE OF FIELD?
0156  7E                 MOV   A,M           GET DEFLECTOR IN CASE AT FIELD EDGE
0157  CA3101             JZ    CHGMOT        IF AT FIELD EDGE — CHG DIRECTION AGAIN
015A  C9                 RET
                * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
                *  THIS ROUTINE CHECKS FOR EDGE OF FIELD, REVERSES MOTION AND
                *       SETS ZERO FLAG IF AT EDGE
                *
015B  78        EDGCK    MOV   A,B           LOAD MOTION DIRECTION FOR TESTING
015C  FE10               CPI   RIGHT         IS MOTION TO RIGHT?
015E  C26801             JNZ   DF060         JMP IF NO
0161  3E3F               MVI   A,X'3F'       FOR RIGHT EDGE HORZ= X'3F'
0163  BB                 CMP   E
0164  C0                 RNZ                 RETURN IF NOT AT RIGHT EDGE
0165  0680               MVI   B,LEFT        REVERSE DIRECTION TO LEFT
0167  C9                 RET
0168  FE20      DF060    CPI   UP            CHECK FOR MOTION UP
016A  C27301             JNZ   DF070         JMP IF MOTION NOT UP
016D  AF                 XRA   A
016E  B2                 ORA   D             CHECK FOR TOP — VERT = 0
016F  C0                 RNZ                 RETURN IF NOT AT TOP
0170  0640               MVI   B,DOWN        REVERSE DIRECTION TO DOWN
0172  C9                 RET
0173  FE40      DF070    CPI   DOWN          CHECK FOR MOTION DOWN
0175  C27F01             JNZ   DF080         JUMP IF MOTION NOT DOWN
0178  3E0F               MVI   A,X'0F'       CHECK FOR BOTTOM — VERT = X'0F'
017A  BA                 CMP   D
017B  C0                 RNZ                 RETURN IF NOT AT BOTTOM
017C  0620               MVI   B,UP          REVERSE DIRECTION TO UP
017E  C9                 RET
017F  AF        DF080    XRA   A             MUST BE MOTION TO LEFT
0180  B3                 ORA   E             CK FOR LEFT EDGE — HORZ = 0
0181  C0                 RNZ                 RETURN IF NOT LEFT EDGE
0182  0610               MVI   B,RIGHT       REVERSE DIRECTION TO RIGHT
0184  C9                 RET
                * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
                *  THIS ROUTINE PERFORMS A TIME DELAY WHILE CONTINUOUSLY
                *            CHECKING FOR KEYBOARD INPUT — INPUT RETURN IN C REG
                *
0185  0E00      DELAY    MVI   C,0           INITIALIZE INPUT SAVE REG
0187  D5                 PUSH  DE            FREE UP D AND E REGS
0188  110007             LXI   DE,X'0700'    LOAD DELAY VALUE                  --1
018B  1B        DL01     DCX   DE            DECREMENT DELAY COUNT
018C  DB06               IN    STATUS        GET KEYBOARD STATUS               --3
018E  E602               ANI   STROBE        CHECK STROBE                      --3
0190  C29801             JNZ   DL02          JMP IF NO  INPUT                  --3
0193  DB05               IN    DATA          GET KEYBOARD DATA                 --3
0195  E67F               ANI   X'7F'         REMOVE BIT 7
0197  4F                 MOV   C,A           SAVE INPUT IN C REG
0198  7A        DL02     MOV   A,D           CHECK FOR
0199  B3                 ORA   E                    TIMER = ZERO
019A  C28B01             JNZ   DL01
019D  D1                 POP   DE            DELAY OVER, RESTORE D AND E REGS
019E  C9                 RET
                * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
                *  THIS ROUTINE INITIALIZES THE VDM PORT AND CLEARS THE SCREEN
                *
019F  AF        CLSCRN   XRA   A
01A0  D3C8               OUT   VDMPORT       CLEAR VDM PORT                    --3
01A2  2100CC             LXI   HL,VDMRAM     POINT TO START OF VDM RAM         --3
```

```
01A5    3EDO                        MVI    A,VDMPAGE+4         A REG = END OF VDM RAM          --3
01A7    3620        CL01            MVI    M,X'20'             BLANK LOCATION GIVEN BY HL
01A9    23                          INX    HL
01AA    BC                          CMP    H                   DONE?
01AB    C2A701                      JNZ    CL01                JMP IF NOT
01AE    C9                          RET

                    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
                    *  THIS ROUTINE DISPLAYS MESSAGE POINTED TO BY BC REGS
                    *
01AF    111DCF      SHOW            LXI    DE,VDMRAM+X'031D'   POINT TO DISPLAY LOCATION       --3
01B2    2607                        MVI    H,7                 MESSAGE LENGTH FIXED AT 7
01B4    0A          SH01            LDAX   BC                  GET BYTE OF MESSAGE
01B5    12                          STAX   DE                  DISPLAY IT
01B6    13                          INX    DE                  BUMP TO
01B7    03                          INX    BC                      NEXT BYTE
01B8    25                          DCR    H                   DECREMENT COUNT
01B9    C2B401                      JNZ    SH01                DONE?
01BC    C9                          RET
                    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
                    *  THIS ROUTINE WAITS AWHILE AND THEN  REMOVES MESSAGE
                    *
01BD    1620        WAER            MVI    D,32                LOAD UNITS OF DELAY             --1
01BF    CD8501      WDLY            CALL   DELAY               WAIT ONE DELAY UNIT
01C2    15                          DCR    D                   CHECK IF
01C3    C2BF01                      JNZ    WDLY                    DONE WAITING
01C6    01DA01                      LXI    BC,BLNK             POINT TO ALL BLANKS
01C9    C3AF01                      JMP    SHOW                DISPLAY BLANKS TO ERASE AND RETURN
                    *
                    *  MESSAGES
                    *
01CC    57494E4E    WIN             DC     X'57494E4E'            'WINNER' MESSAGE
01D0    455220                      DC     X'455220'
01D3    4E4F2054    LOSE            DC     X'4E4F2054'            'NO TIME' MESSAGE
01D7    494D45                      DC     X'494D45'
01DA    20202020    BLNK            DC     X'20202020'            (BLANKS)
01DE    202020                      DC     X'202020'
                    *
                    *  SAVE AREAS AND STACK
                    *
01E1                TIMER           DS     CL2                 TIMER SAVE AREA
01E3                TARNUM          DS     CL1                 NUMBER OF TARGETS SAVE AREA
01E4                STACK           DS     CL8                 STACK AREA (ONLY 4 DEEP)
                    *
                                    END
```

R. W. Burhans
E.E. Dept. Ohio Univ.
Athens OH 45701

KIM Experimenters Breadboard. The decoder is fastened to the upper left of the KIM, and the WASATCH boards are to the far left. (Courtesy Ohio University publications.)

How Much Memory for a KIM?

is 28K enough?

This decoder is designed for use with WASATCH compact low-power 4K memory boards, which are available in kit form for $79.95 (WASATCH Semiconductor Products, 25 South 300 East, Suite 215, Salt Lake City UT 84111). These boards need a positive board select signal (B/S). The lower 4K (already decoded on KIM as K1, 2, 3, 4) is wire ORed to an inverter to provide B/S 1 (0400-13FF). A 74LS138 decoder chip disables the D input when addresses above 1FFF are used with a simple diode OR to the D enable on KIM. The highest address decoded is 7XXX, which is wire ORed to the K7 input on KIM, returning the system to the KIM monitor when an interrupt RST occurs for all 1s on the address bus. In my system, a motherboard is hardwired for B/S 1, 2, 3 slots. The other outputs from the decoder at $\overline{B/S}$ 4, 5, 6, could be used as unlatched memory-mapped output ports, or they can be inverted and wired as B/S for additional 4K WASATCH boards.

One more 4K slot, 7, could be used as $\overline{B/S}$ 7 if desired by tying AB15 (KIM E-T) as the interrupt vector to K7 through an inverter and diode. This decoding scheme ignores address line AB15, making the KIM effectively a 32K machine when the appropriate 4K slots are filled. In the original KIM, address lines AB13, AB14 and AB15 were not used, making it a potential 8K machine.

The same decoder could also be used with 8K memory boards by wire ORing two adjacent B/S lines together with diodes as in the 0 and 1 output lines in the circuit wire ORed to D. These diode OR circuits work just as well as open-collector OR circuits and save the chip count. The entire decoder can be placed on a small 1 x 2 inch board fastened to the upper left corner of the KIM. ■

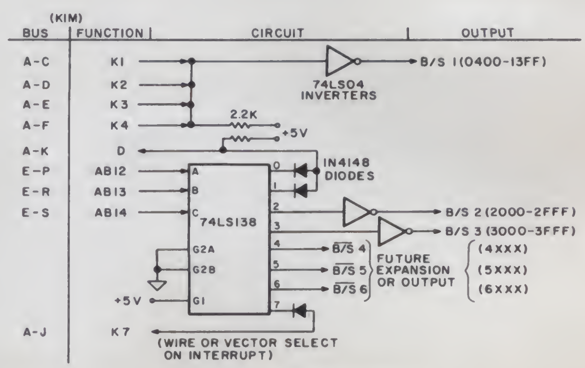
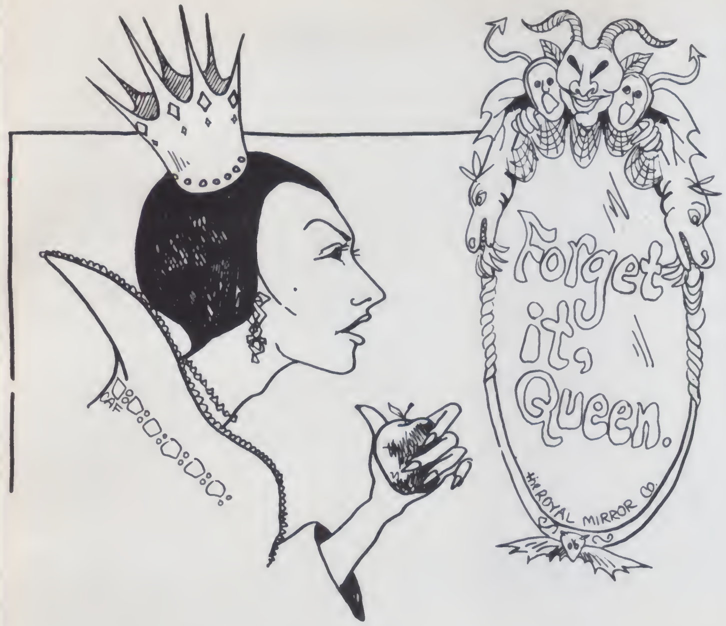


Fig. 1. Decoder schematic.



"What do you mean—Snow White's biorhythms look more auspicious than mine?!!"

LEGAL/ BUSINESS FORUM

(from page 22)

major causes of repairs were poor construction of products that were sold as kits and component failure in preassembled units.

None of the manufacturers reported that they had been involved in a lawsuit as a result of warranty problems. Perhaps this is attributable to the short time the manufacturers have been in existence. On the other hand, it might also be attributable to the tendency of some computer hobbyists to sit back and complain about the failure to receive warranty service rather than taking any action to do something about it.

What Should the Hobbyist Do

The hobbyist can best help the situation by becoming aware of his rights and by enforcing them. When you go into a store to purchase a product, ask to see the written warranty that accompanies it. Also ask the salesman about the individual store's own warranty policy. If the salesman cannot provide you with the written warranty, inform him of the requirements of the Presale Availability Rule. If he is mealymouthed about the store's own warranty policy, ask to talk to the store owner. If *he* is mealymouthed, take your business elsewhere.

If you find that a retailer or manufacturer is not conforming to the rules, write to the Federal Trade Commission (FTC). There are many regional offices; your local consumer-protection office or complaint center will be able to tell you where the nearest office is. You can find the local consumer-protection office under the name of your city, county or state government in the phone book.

The FTC cannot help you directly with a warranty problem. But it needs to know if companies are obeying the warranty rules so as to enforce them if they are not. If you have sent a board into a manufacturer to be repaired and have not gotten it back within a reasonable amount of time—perhaps a month—write a letter to the manufacturer complaining about the problem. Indicate on the face of the letter that a copy is going to the FTC. If you still don't receive satisfaction, see your attorney. Remember, you can recover attorney's fees and court costs in addition to damages if you are successful.

I can't emphasize it enough. You must take advantage of your rights or you will be the one who is taken advantage of.

BOOKS

(from page 16)

the author early in the first chapter.

". . . It may be noted that the symbolic language used throughout this book is a combination of the original mnemonics for the instruction set of the 8008, which is a subset of the 8080 instruction set, plus a version of the mnemonics presented by Intel Corporation for the additional instructions of the 8080. For those readers who may be familiar with the mnemonics as presented by Intel Corporation, Appendix A lists the mnemonics used here along with the equivalent Intel mnemonics."

The only resemblance between the mnemonics for the 8008 and the 8080 is in the description of the command's function. The 8008 is not a subset of the 8080 instruction set in the sense that we think of the 8080 as a subset of the Z-80. In the 8008, both the mnemonic and octal or hex representations of the command are different from those of the 8080. When the author discusses the "8080 instruction set," 48 of the 78 instructions are actually 8008 instructions. He presents the mnemonic plus the octal and hex representation of the command, and you must then turn to the appendix to find out what the 8080 mnemonic is.

This problem is compounded when you get into the routines. One of the fastest ways for me to learn a programming language is by studying routines and pro-

grams written in the language and properly documented with comments. Example 1 is a routine from the book. The 8080 equivalent routine is shown in Example 2.

There are some good discussions on programming techniques and instruction descriptions in this book. However, the reader must look up all the instructions in the appendix, which is frustrating, and the author should have used the accepted Intel 8080 mnemonics throughout the book.

Jan A. Heise
Montgomery AL

```
MOVEAD,     LAM             Fetch contents to be moved.
            STAD            Store contents in new location.
            LAH             Fetch high portion of FROM address.
            CPB             Is FROM page = page limit?
            JFZ MOVAD1      No, continue transfer.
            LAL             Yes, fetch low part of FROM address.
            CPC             Is FROM low = low address limit?
            RTZ             Yes, return to calling program.
MOVAD1      INXH            No, advance FROM pointer.
            INXD            Advance TO pointer.
            JMP MOVEAD      Continue with next transfer.
```

Example 1.

```
MOVEAD      MOV A,M
            STAX D
            MOV A,H
            CMP B
            JNZ MOVAD1
            MOV A,L
            CMP C
            RZ
MOVAD1      INX H
            INX D
            JMP MOVEAD
```

Example 2.

A Collection of Programming Problems and Techniques
H.A. Maurer and M.R. Williams
Prentice-Hall, 1972, $6.95

Being one who believes in the "learn by doing" approach to education, I found the contents of this paperback fulfilling. Introductory books offer too few practical problems, while more complex texts tend to be theoretical. However, the authors of this book have struck a good compromise between these two extremes.

I've read this book twice; the first time was when I considered myself a pure beginner in computer programming. I'm not an expert yet, but experience has sharpened my gray matter so I can better appreciate the problems and techniques described in this book.

If you look for the standard BASIC solution-type program listings in this book, you won't find them. The authors present problems and suggested solutions, not program listings that solve the problems.

The book is neatly laid out: A problem is presented, and the reader is left to his own devices to find the solution. Since you're the programmer, it's up to you to code the problem and solve it using techniques given in the book or by calling upon your own methods. Selected problem solutions are in the back of the book for those who wish to check their answers. The solutions are given in a common sense or, where applicable, a mathematical approach.

Novices can use this text to gain experience and insight into the mechanics of problem solution.

Likewise, the expert can profit by applying past experience in solving the problems. Of course, all of us benefit by being exposed to a wide range of problems that are readily solved by a computer program.

Much of the book is devoted to mathematical problems, such as: equations in one variable (chapter 6); random numbers (chapter 8); statistics (chapter 12); and systems of linear equations (chapter 13). High-school mathematics is sufficient to solve these problems, so the non-mathematician need not be frightened by these chapters.

No programming book is complete without at least one chapter devoted to games. Although not game-oriented, this book does introduce a few problems dealing with games.

Two chapters present problems concerned with merging (chapter 2) and sorting (chapter 7). There is good introductory material here, and the serious reader will want to study other texts dealing with these two important aspects of programming.

Plotting (chapter 14) should whet the appetite of the graphics people, and chapter 9 should assist those interested in input/output problems.

This book is worth having on your bookshelf. For the serious computer user, hobbyist or professional, the problem-and-solution approach teaches by presenting a problem and allowing the reader to finalize a solution based on his particular level of knowledge and experience. With about 400 problems in this book, there must be at least a dozen that should interest you.

Len Gorney
Clarks Summit PA

NEW PRODUCTS

(from page 15)

tape at 50 characters per second, while the punch operates at a rate of ten cps. The punch and reader circuits are completely independent of one another and may be operated simultaneously.

Features of the H10 include a copy mode for tape duplication, a built-in heavy-duty power supply and a stepper motor for reliable reader tape drive. The interface is standard parallel TTL. The mail-order kit price is $350.

Heath Company, Dept. 350-26, Benton Harbor MI 49022.

TDL Video Display Board

Technical Design Labs has introduced a video interface for S-100-bus microcomputers. The Video Display Board (VDB) is low cost and provides the capabilities of video terminals costing much more. It consists of two boards, one piggybacked to the other. The unit occupies one edge connector on the bus, but takes up the space of two boards.

The VDB contains its own display buffer memory and provides two pages of display, each with 25 rows of 80 characters. The display buffer memory does not use any memory address, thus leaving the entire computer memory address intact for user programs.

In addition to the 96 uppercase and lowercase ASCII characters with decenders, this product displays 64 unique display symbols, thus permitting a graphic resolution with 160 horizontal elements by 75 vertical elements. The display can accept data at a 400,000 character-per-second rate.

The board works with either modified TV set or monitor and has an on-board 8-bit parallel keyboard port with status strobes. The VDB requires one motherboard socket and occupies

two card spaces. The unit is priced at $349 in kit form and $449 assembled and tested. Software character and graphics output drivers for Z-80 and 8080 systems are supplied. These drivers are ROMable.

Technical Design Labs, Research Park, Building H, 1101 State Road, Princeton NJ 08540.

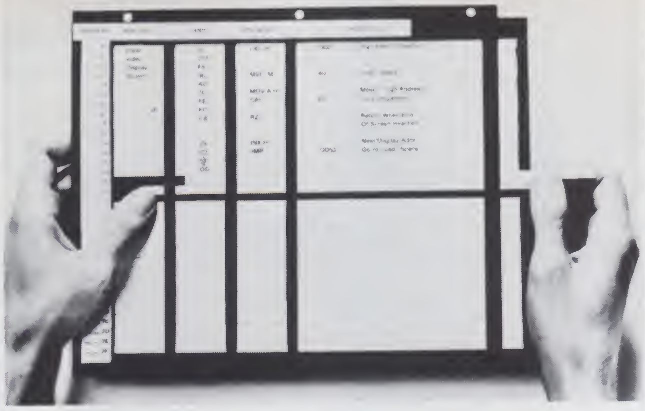
6800 Home Study Course

With the purchase of a basic Micro-68a microprocessor you receive a complete home-study course, including User's Manual, 15-chapter Lab Manual, Understanding Microprocessors, M6800 Design Manual, the Motorola Programming Manual and Instruction Summary Card. The total cost is $544.50.

Electronic Product Associates Inc., 1157 Vega Street, San Diego CA 92110.

Vamp Programming Aid

The BOPA (Basic Operational Programming Aid) is a microcomputing aid that will help you write programs faster and more accurately. It comes with 32 removable slats, on which you write memory entries with a special ink pen. Once your program is written on the slats, you can edit, modify, rearrange and insert instructions by simply moving the slats around. Memory addressing is always current and automatically updates when you make any program changes, thus making assembly and compiling of your programs quick and easy.



Vamp BOPA.

The BOPA is an excellent learning aid for beginners. Memory mapping is simplified, and corrections are easily made by reversing the slats or by erasure. A complete set of boards can be carried in a three-ring binder. Single-card system, with 32 entries and 32-byte address loop, $11.95; four-card system, with 128 entries and 256-byte address loop, pen and Solvex solution, $39.95; complete system, with 256 entries and a 256-byte address loop, pen and Solvex solution, $74.95.

Vamp, Inc., PO Box 29315, Los Angeles CA 90029.

Music System Program

Software Technology Corporation's Music System comes complete with a program on cassette tape, six sample selections, a user's manual and a circuit board with components. The program is priced at $24.50.

Running in about 2K of memory, the program includes a monitor, text editor with ALS-8-compatible file structure, and a high-level music-composing language compiler. Language capabilities include dotted notes, four-octave range and staccato. The S-100-bus-compatible circuit board can be assembled in a few minutes.

The program generates music by producing three simultaneous tones of fixed amplitude with a complex waveform that approximates a reed organ. The Music System generates tones using square waves, produced by a highly controlled pulsing of one of the S-100-bus status lines. With the addition of amplifier, speaker, cable and any S-100-bus computer the Music System is ready to play.

Software Technology Corp., PO Box 5260, San Mateo CA 94402.

KB CALENDAR

Long Beach CA

PERCOMP '78 (co-sponsored by the International Computer Society/SCCS and the Rockwell Hobbyist Computer Club) will be held at the Long Beach Convention Center, Long Beach CA, April 28-30, 1978. PERCOMP is a selling show designed with the home computerist and small-business person in mind. For information concerning seminars, contact James Lindwedel, Technical Program Chairperson, PERCOMP '78, 1833 E. 17th St., Santa Ana CA 92701.

Lexington VA

A two-week course in digital-electronics and microcomputer-interfacing fundamentals will be held at Virginia Military Institute from July 17 through July 29, 1978. For information and registration forms write to: Dr. Philip B. Peters, Dept. of Physics, VMI, Lexington VA 24450.

San Francisco CA

COMPCON '78, Jack Tar Hotel, San Francisco, February 28-March 2, 1978, (starting at 7 pm) will look at the phenomenon of personal computing. Four panel sessions have been arranged with experts who will be discussing various aspects of the computer revolution. The conference registration fee covers attendance at all personal computing sessions and exhibits. There is a registration fee of $5 for individuals wishing to only attend the personal computing sessions and exhibits. Contact organizers Alice Ahlgren, Mountain View CA, (415) 964-7400 and/or Robert Albrecht, Menlo Park CA, (415) 323-6117, for more information.

Anaheim CA

A Personal Computing Festival will share the public spotlight with the 1978 National Computer Conference June 5-8 in Anaheim. CA. A call for papers has been issued for the Festival Program, to be held June 6-8 at the Disneyland Hotel adjacent to the Anaheim Convention Center. The three-day program will include presentations of invited papers, contributed papers and tutorials, as well as panel discussions relevant to personal computing. Letters of intent to participate as either an author, panelist or session chairman must be submitted by February 1, 1978. Authors who have received notification of acceptance must submit final papers by March 15, 1978 in a specified camera-ready format. Information on NCC '78 may be obtained from AFIPS, 210 Summit Avenue, Montvale NJ 97645, or by calling (201) 391-9810

Atlanta GA

Papers are invited for presentation at the 16th Annual Convention of the Association for Educational Data Systems, Atlanta GA, May 15-19, 1978. Papers are solicited in all categories of educational use of computers. Papers submitted will be reviewed by a panel, and authors of those accepted will be invited to present their papers and to have them published in the proceedings.

Judges will select an outstanding paper from each category, and a panel will select a best paper on the basis of content, presentation and overall quality to receive a $500 cash award. For further information, contact: Dr. James E. Eisele, Office of Computing Activities, University of Georgia, Athens GA 30602.

Kilobaud Classified

Kilobaud classified advertisements are intended for use by those individuals desiring to buy, sell, or trade used computer equipment or software. No commercial ads are accepted.

Two sizes of ads are available. The $5 box allows five lines of about 22 characters each, including spaces and punctuation. The $10 box provides ten lines of type – again, each line is about 22 characters. Minimize capital letters, as they use twice the space of small characters. Payment is required in advance with ad copy. We cannot bill, or accept credit. Oversize ads are not accepted. Each subscriber is limited to two (2) identical ads in any given issue.

Advertising text and payment must reach us 60 days in advance of publication. For example, advertising copy for the March issue (mailed in February) must be in our hands on January 1. The publisher reserves the right to refuse a questionable or not applicable advertisement. Mail advertisements to: KILOBAUD CLASSIFIED, Kilobaud, Peterborough, NH 03458. Do not include any other material with your ad, as it may be delayed.

For Sale: Federal Income Tax program in BASIC; runs in 8K. Prints results on form 1040. Source listing $14.70. PET or TRS-80 tape for $16.50 ea. C.R. Lufkin, 315 Dominion Dr., Newport News VA 23602.

PET-2001 and Radio Shack TRS-80 arrived on campus. I want to survey users and report results to any interested hobbyists. Write: Professor Bill Parks, Walters State Community College, Morristown TN 37814.

Friden 8 unit with card reader punch, $280; Burroughs Selectric, $380; 2 Model 15 Baudot, $50 ea. Ivan Whitehouse, Box 206, Goldendale WA 98620.

THIS is a $5 ad. If desired, ads may cross a $5 or $10 boundary. Rates increase in $5 increments only. Be sure to enclose sufficient payment!

For sale: OSI Challenger, 12K RAM, video board, cassette interface, TTY port, many extra boards, all manuals MOS and OSI, keyboard. Software: 8K BASIC, 4K BASIC, extended monitor. Much more. Must call for full details. $1200 takes all, or call for separate pricing. Just plug in TV and go. Mark Gilger, 3306 S. Maple, Rapid City SD 57701, (605) 342-4190.

For Sale: 2 ea. 4K DYN, 2 ea. 4K STATIC, $85 ea; SWTPC CT-1024 CRT w/cabinet, $250; all excellent with sockets. R. White, 8530 Stonehaven, Boise ID 83704. Call (208) 377-0336 after 5.

Software: North Star Disk Media. Startrek $2, Checker $2, UFO $2, Family Budget $5, Poker & Craps $3, Decision Helper $4. Send for complete list. Software, Box AF, Woodbridge CT 06525.

Contest!

Well, the votes have been counted for the best article in the November issue . . . and the winner is Ron Lange, author of "Build the $35 Modem," appearing on page 94 of *Kilobaud* No. 11. Ron will soon be receiving a check for $100. Congratulations, Ron!

In a drawing of all those votes submitted for November's article winner, the card of Ted A. Heider of Parma OH was selected. Ted's prize is a choice of any book we publish.

Next month's issue will bring the winner for December. After we've had a chance to tally all the votes submitted for 1977, the winner of the best article for the year will be selected.

To all readers: Thanks for your votes; and keep voting!

LEARN TO PROGRAM MICROCOMPUTERS

And at an affordable price. The Modu-Learn™ home study course from Logical Services.

Now you can learn microcomputer programming in ten comprehensible lessons. At home. In your own time. At your own pace.

You learn to solve complex problems by breaking them down into easily programmed modules. Prepared by professional design engineers, the Modu-Learn™ course presents systematic software design techniques, structured program design, and practical examples from real 8080A micro-computer applications. All in a modular sequence of 10 lessons . . . more than 500 pages, bound into one practical notebook for easy reference.

You get diverse examples, problems, and solutions. With thorough background material on micro-computer architecture, hardware/software trade-offs, and useful reference tables. All for only $49.95.

For $49.95 you learn design techniques that make software work for you. Modu-Learn™ starts with the basics. Our problem-solution approach enables you to "graduate" as a programmer.

Circle the reader service number below to receive our free descriptive brochure and course outline.

Use your Master Charge or VISA card to order today. Call Pat at (415) 965-8365.

711 Stierlin Road
Mountain View, CA 94043
(415) 965-8365

L3

LOGICAL
SERVICES INCORPORATED

RAINBOW COMPUTING INC.

Supplier of
Southwest Technical Products
Apple
Wave Mate
The Digital Group
Digital Equipment Corporation
Computer Products

Peripherals and Supplies from

PerSci	**Computer Devices**
Contronix	**Lear-Siegler**
Diablo	**Multi-Tech**
Maxell	**Texas Instruments**

'Scotch' Brand Magnetic Media
Specialists in Design, Implementation and Support of Custom Hardware/Software for Business, Educational, and Personal Use
Consulting/Contract/Programming
Operating Systems/Applications Software

Experts in most major computer software including
CDC, IBM, PDP
BASIC, COBOL, FORTRAN, PL1
Lisp, Simula, Snobol, SPSS, BMD's
COMPASS, MACRO, 6800, & Z80 assembly languages

10723 White Oak Ave., Granada Hills, Ca. 91344
(213) 360-2171

R10

MICROCOMPUTERS, INC.

SMALL BUSINESS SYSTEMS
DEDICATED MICROPROCESSOR
APPLICATIONS

539 Amherst St.
Nashua, N.H. 03060 Tel. (603) 889-1646

M16

GENERAL LEDGER
with
INCOME STATEMENT
& BALANCE SHEET

For any computer using the North Star micro-disk system

Program disk $70
Documentation $5

Available Now

THE COMPUTER ROOM

690 Yellowstone Ave.— Suite D
POCATELLO, IDAHO 83201

C59

MAN/PAK HORIZON-2 COMPUTER

ASSEMBLED, READY TO PLUG INTO PRINTER, CRT... & GO!

ONLY $2395

MAN/PAK HORIZON-2 COMPUTER
uses a full speed (4 MHz) Z-80 microprocessor, the (S-100) bus. & includes 16K byte memory with parity option, a disk controller with two Shugart minifloppy disk drives, & full extended disk BASIC, plus a serial & parallel I/O port, 6 edge connectors & wood cover.

IMMEDIATE DELIVERY!
Complete computer system including 32K memory, Hazeltine 1500 Video Terminal & Centronics 779 Serial Printer, all assembled . . . READY-TO-GO! Only $5145 Complete.

Send Check, M.O.
or for more information to:

M21

MANCHESTER EQUIPMENT CO.
120 Bethpage Road
Hicksville, N.Y. 11801
Call Collect: (516) 433-0613
Dealer Inquiries Invited

THE BIT STOP

... Announces ...

THE CASSETTE INTERFACE KIT

FEATURES:

1. Z80 – 8080, 6800 and 6502 system compatibility . . . requires only a one-bit input port and a four-bit output port (including two bits for optional status indicators).
2. Software-oriented system – your computer does the timing.
3. Standard speed 350 bits per second (44 bytes per second), easily converted to higher speeds (software changes only!).
4. Thoroughly tested on a variety of inexpensive recorders.
5. Complete 11-page manual plus thoroughly documented software. Included are a complete description of hardware, startup hints, timing charts, etc.
6. Special introductory kit price of $29.95. Price $39.95 after December 31, 1977. (Add $5.00 for wired and tested unit.) Please specify Z80–8080, 6800 or 6502 when ordering.

THE BIT STOP
P.O. Box 973
Mobile, Alabama
36601

B21

Popular products, at proper prices.

CMOS

4000	$0.25
4001	0.29
4002	0.34
4007	0.29
4008	1.28
4009	0.53
4010	0.53
4011	0.29
4012	0.29
4013	0.50
4014	1.23
4015	0.90
4016	0.45
4017	1.23
4019	0.55
4020	1.50
4021	1.23
4022	1.20
4023	0.29
4024	1.03
4025	0.29
4027	0.75
4028	1.00
4029	1.73
4030	0.53
4033	1.50
4037	0.50
4040	1.50
4041	0.85
4042	0.85
4043	0.60
4044	0.60
4047	1.63
4049	0.50
4050	0.50
4051	1.03
4052	1.03
4053	1.03
4060	1.48
4066	0.58
4069B/74C04	0.33
4070	0.60
4071	0.33
4073	0.33
4075	0.33
4076B/74C173	1.63
4081	0.33
4116	0.50
14511	2.00

74LS TTL

74LS00	0.30
74LS01	0.30
74LS02	0.30
74LS04	0.33
74LS08	0.36
74LS10	0.30
74LS11	0.36
74LS12	0.33
74LS14	1.38
74LS15	0.30
74LS20	0.30
74LS21	0.33
74LS22	0.33
74LS26	0.43
74LS27	0.36
74LS30	0.30
74LS32	0.38
74LS37	0.45
74LS38	0.45
74LS42	0.98
74LS47	1.00
74LS48	0.98
74LS74	0.50
74LS75	0.68
74LS76	0.50
74LS86	0.50
74LS109	0.50
74LS125	0.63
74LS126	0.63
74LS132	1.25
74LS138	1.10
74LS139	1.15
74LS151	0.95
74LS155	1.38
74LS157	0.95
74LS160	1.40
74LS161	1.40
74LS162	1.40
74LS163	1.40
74LS168	1.87
74LS169	1.87
74LS173	1.65
74LS174	1.25
74LS175	1.15
74LS240	1.88
74LS257	1.25
74LS258	1.25
74LS266	0.53
74LS283	1.20
74LS365/ 80LS95	0.75
74LS366/ 80LS96	0.75
74LS367/ 80LS97	0.75
74LS368/ 80LS98	0.75
74LS386	0.55
81LS95	1.13
81LS96	1.13
81LS97	1.13
81LS98	1.13

TTL

7400	0.18
7401	0.20
7402	0.20
7403	0.20
7404	0.21
7405	0.22
7406	0.33
7407	0.36
7408	0.23
7409	0.23
7410	0.18
7411	0.22
7413	0.45
7414	1.05
7416	0.30
7417	0.33
7420	0.18
7422	0.23
7425	0.30
7426	0.25
7427	0.29
7430	0.26
7432	0.30
7437	0.36
7438	0.36
7439	0.45
7440	0.19
7442	0.55
7444	0.45
7445	0.63
7446	0.85
7447	0.80
7448	0.83
7450	0.18
7454	0.18
7460	0.18
7472	0.29
7473	0.36
7474	0.36
7475	0.65
7476	0.36
7483	0.89
7485	0.99
7486	0.36
7489	1.75
7490	0.60
7492	0.63
7495	0.90
7496	0.90
74107	0.36
74109	0.53
74121	0.45
74123	0.65
74125/8093	0.50
74126/8094	0.53
74128	0.65
74145	0.63
74148	1.25
74150	1.07
74151	0.71
74153	0.71
74154	1.23
74155	0.71
74156	0.90
74157	0.71
74159	2.20
74160	0.89
74161	0.89
74162	0.89
74163	0.89
74164	1.34
74165	1.34
74173	1.34
74174	1.52
74175	1.34
74176	0.89
74177	0.88
74179	0.75
74180	0.98
74181	2.23
74182	0.80
74190	1.00
74191	1.00
74192	1.34
74193	1.34
74194	1.34
74195	0.71
74196	0.98
74197	0.98
74198	1.96
74199	1.96
74273	1.89
74S287/ 6301/ 82S129	4.38
74S288/8223	2.50
74365	0.69
74366	0.69
74367	0.69
74368	0.69
74390	1.50
74393	1.50
74S471	7.50
75325	0.65
75492	0.80

SPECIALS

¼" phone jacks: mono, closed circuit.

The real thing---Switchcraft---for less than the imported kind:

10/$3.00

Cable ties: can accomodate bundles of wire or cable from 1/16" to 3/4" in diameter.

100/$2.95

MEMORIES

ECONORAM II™

This is the 8K X 8 static memory that started it all...plug it into your Altair, IMSAI, or other S-100 buss computer, and you've got a memory that just goes on working and working and working. Configured as 2 independent 4K blocks, with separate protect for each block and vector interrupt provision if you try to write into protected memory. Handles DMA devices. All address and data lines fully buffered. Tri-state outputs for use with bi-directional busses. Selectable write strobe (writes on either PWR or MWRITE), and dip switch selectable address. We guarantee 450 ns speed, although many users report running this board in conjunction with 4 MHz Z-80s without using the on board wait state provision. The mechanical quality matches the design, with gold-plated edge fingers, legended and solder masked board, sockets for all ICs, and industrial grade or better components. Join the thousands who have made this our most popular board!

Kit form....................................$135.00
Assembled, tested, warranted for 1 year.......$155.00
Special! 3 kits (24K of memory)............3/$375.00

ECONORAM III™

Here is the first 8K **dynamic** RAM that performs well enough to merit the ECONORAM trade mark. Thanks to the "SynchroFresh"* timing process, refresh fits naturally into the timing of the S-100 buss...now you can have half the power of statics, but without the traditional timing hassles you've come to associate with dynamic memories.

In addition to low power, this board runs at zero wait states with an 8080 CPU, and is configured as 2 separate blocks for maximum versatility. **Not a kit:** shipped assembled, tested, and ready to plug into any S-100 buss computer (Altair, IMSAI, etc.). 1 year warranty, parts and performance..............**$149.00**

*SynchroFresh is a trade mark of Thinker Toys

ECONORAM VI™

We proudly welcome our newest memory board family member, designed from the ground up for full compatibility with the Heath Company H8. Organized as 2 independent blocks for flexibility: one 8K block (locatable on any 8K boundary) and one 4K block locatable on any 4K boundary. Has the same basic features of our **ECONORAM II**tm---all static design, dip switch address selection, switch selected write protect and phantom, sockets for all ICs, full buffering---plus the required hardware, and edge connector, to mate mechanically with the H8. Now you can have our 12K board for the price of the Heath Company's 8K...with the performance you've come to expect from products carrying the ECONORAM trade mark. **Kit form: $235.00**

Also available: H8 4K to 8K conversion kit (**$90**). If you have a Heath Company 4K memory, plug in these ICs and you'll have a full 8K. Kit includes eight TI4044 memories and matching sockets.

Send for our flyer

TERMS: Please allow up to 5% for shipping; excess refunded. Californians add tax. COD orders accepted but we must have street address for UPS. For VISA* or Mastercharge* orders ($15 minimum) call our 24 hour order desk at **(415) 562-0636**. Prices good through cover month of magazine. Thank you!

GODBOUT

BILL GODBOUT ELECTRONICS
BOX 2355, OAKLAND AIRPORT, CA 94614

G4

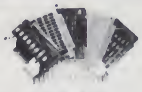
ADVANCED COMPUTER PRODUCTS

THE FIRST TO OFFER PRIME PRODUCTS TO THE HOBBYIST AT FAIR PRICES NOW LOWERS PRICES EVEN FURTHER!

1. Proven Quality Factory tested products only, no re-tests or fallouts. Guaranteed money back. We stand behind our products.

2. Same Day Shipment All orders will be shipped same day as received.

MOTOROLA 6800 COMPATIBLE MODULES



MEK6800 D2 KIT $235.00

9601 16 Slot Mother Bd.	$175.00
9602 16 Slot Card Cage	72.00
9610 Proto Board	36.00
9615 4K EPROM Module (1702A)	350.00
9615K 4K EPROM Kit Module	275.00
9620 16 Port Parallel I/O	375.00
9626 8K Static RAM Module	350.00
9626K 8K Static RAM Kit	275.00
9650 8 Port Duplex Asyn Serial I/O	395.00
Connectors $6.50	16/72.00

All assembled & tested not Kits

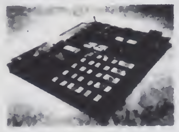
PLUS MOTOROLA TV MONITORS-PRIME

Model M3003-L01 12" display	$219.95
Model M2000-155 9" display	$199.95

Add $10.00 for shipping

AMI EVK 99

only $133.00



EVK 99 Kit	$133.00
EVK 99 EX expanded	449.95
Universal Kluge Board	95.00
16K Byte RAM Board	75.00
6 Slot Motherboard	35.00
Extender Board	45.00
Video Board	95.00
Connectors . . $6.50	12/59.95
Solid Frame Chassis	120.00
Frame Chassis	69.95
Micro Assembler ROM	30.00
Proto ROM	30.00
Tiny Basic Papertape	20.00
Tiny Basic EPROM	125.00

RAM'S (cont.)

4096 16 pin	4.95
2107B 4	4.25
MM5261	3.95
MM5262	1.95
MM5280	4.95
8599	1.88
P3101	2.95
5101C-E	9.95
3107	3.95
1101	1.95
1103	1.95
4008 L	2.95

FLOPPY DISK

MPD 372	$55.00
WD1771 Floppy	59.95

Application notes included

CHARACTER GENERATORS

R032513 Upper	$10.95
R032513 Lower	9.95
HD0165	9.95
MC6571	14.95
MC6574	14.95
MC6575	14.95

KEYBOARD ENCODERS

AY52376	$14.95
AY53600	14.95

UARTS/USRTS

AY51013A (5V)	$ 5.50
TR1602B (5V)	5.50
AY51014 (5-14V)	8.95
AY51015 (5V)	9.95
IM6402 (5V)	12.95
IM6503 (5V)	12.95
S2350	10.95
WD1671B Astros	29.95

BAUD RATE GENERATORS

MC14411	$11.95
WD1941 Dual	9.95
34702	13.95

TV CHIPS

MM5320 TV Synch.	$ 7.95
MM5369 Prescaler	3.95
MM57100 Game Chip	16.95
MM57104 Clock	3.75
LM1889 Modulator	3.95
CW300 Saw Function	1.50
AY38500-1 TV Game	9.95
AY38600-1 Color TV Game	24.95
AY38615-1 Color Converter	8.95
AY38700-1 Tank Chip	29.95

ORGAN CHIPS

MM5554, MM5555, MM5556 . . . NSC Set $19.95

CLOCK CHIPS

MM5313	$4.50
MK 50250	4.50
MM5314	4.95
MM5316	4.95
MM5375	4.95

A TO D CONVERTORS

8700CJ	$13.95
8701CN	21.95
8750CJ	13.95
1408L8	5.95

DISPLAYS/LEDS

FND 357 (.375)	$1.25
FND 367 (.375)	1.35
FND 500 (.500)	1.25
FND 507 (.500)	1.00
FLV 110 Redled	5/1.00
FND 800 (.800)	2.50
FND 807 (.800)	2.50
HP 7730 7 Segment	1.50
HP 7340 Hex Display	15.95

UV EPROM ERASER NEW LOW-COST MODEL ERASES 4 PROMS AT ONE TIME $59.95

SOCKETS

14 Pin ww	.37
16 Pin ww	.38
18 Pin ww	.60
22 Pin ww	.89
24 Pin ww	.85
40 Pin ww	1.25
8 Pin S/T	.17
14 Pin S/T	.20
16 Pin S/T	.22
18 Pin S/T	.31
22 Pin S/T	.35
24 Pin S/T	.41
28 Pin S/T	.49
40 Pin S/T	.63

NEW CTS DIPSWITCHES

CTS206-4	$1.75
CTS206-5	$1.75
CTS206-6	$1.75
CTS207-7	$1.75
CTS208-8	$1.95
CTS209-9	$1.95
CTS209-10	$1.95

NEW CATALOG NOW AVAILABLE

CONNECTORS

S100 BUS W/W IMSAI	$5.00
S100 BUS S/T IMSAI	5.00
S100 BUS W/W ALTAIR	5.50
S100 BUS S/T ALTAIR	5.25
RS232 25 Pin Male DB25P	3.50
PS232 25 Pin Female DB25S	3.95
Plastic Hood	.75
IMSAI Card Guides	4/1.00

RS232 CONNECTOR SET WITH HOOD 6.50

CRYSTALS

1.0MH$_3$	$ 5.85	10.0MH$_3$	4.95
2.0MH$_3$	5.85	13.0MH$_3$	4.95
2.097152MH$_3$	5.85	14.31818	4.95
2.4576MH$_3$	5.85	18.0MH$_3$	4.95
3.579545MH$_3$	3.95	18.432MH$_3$	5.95
4.0MH$_3$	4.95	20.0MH$_3$	4.95
4.194304MH$_3$	5.95	22.1184MH$_3$	5.95
4.91520MH$_3$	5.95	27.000MH$_3$	5.95
5.0MH$_3$	4.95	36.000MH$_3$	5.95
5.7143MH$_3$	5.95	48.000MH$_3$	5.95
6.0MH$_3$	4.95	100KC	12.95

MICROPROCESSORS

Z-80	$24.95
Z-80A	34.95
F-8 (3850)	19.95
2650	24.95
CD1802	19.95
8080A	12.95
8085	29.95
8008-1	14.95
2901	21.95
2901A	29.95
TMS 9900JL	89.95
CP1600	39.95
6502	19.95
IM6100	29.95
6800P	24.95
6802P	32.95

SUPPORT DEVICES

3881 (Z80 PIO)	$12.95
3882 (Z80 CTC)	12.95
3851 (F8 Program	14.95
3853 (F8 Memory I/O)	14.95
8212 8 bit I/O	3.50
8214 Priority Interrupt	9.95
8216 Bus driver	3.75
8224 Clock Gen.	5.95
8224-4 4MHz CLK Gen.	12.95
8T26 Busdriver	2.25

SUPPORT DEVICES (cont.)

8226 Busdriver	3.95
8228 Sys Controller	7.95
8238 Sys Controller	9.95
8251/9551 Prog. Comm. I/O	11.95
8253 Interval Timer	21.95
8255/9555 Prog. Periph. I/O	11.95
8257 DMA Control	32.95
8259 Prog. Interrupt	32.95
6810-1 128 x 8 RAM	4.95
6820 PIA	8.95
6834-1 EPROM 4K	14.95
6834 EPROM 4K	16.95
6850 ACIA	9.95
6852 Serial Adapter	11.95
6860 Modem	14.95
6862 RPS Modulator	17.95
6871B 1.0 MHz Clock OSC	26.95
6880 MPU Busdriver	2.95
68 MinII Minibug 2	49.95
68 MinIII Minibug 3	74.95
6831/10224 Micro Assembler Rom	30.00
6831/11003 Proto Rom	30.00
1821SCD 1K RAM	15.95
1822SCD 256 x 4 RAM	16.95
1824CD 32 x 8 RAM	9.95
1852CD 8 bit I/O	10.95
1856CD I/O	8.95
1857CD I/O	8.95

PROM'S

2708	$19.95
1702A	7.95
1702-6	4.49
5204	16.95
6834	16.95
6834-1	14.95
82S126	4.50
8223	2.95
5203	6.95

Programming Available

RAM'S

21L02 (450ns)	$1.75
21L02 (250ns)	2.25
2102(450ns)	1.69
2102(650ns)	1.59
2102(1 sec)	1.40
2111	3.95
21L11	4.25
2112	2.95
2101	2.95
21L01	3.95
74C89	4.95
74S189	2.95
7489	1.88
4116 16K	34.95
TMS4060 22 pin	4.25
WD4060 22 pin	3.25

LOGOS I THE BEST 8K RAM KIT

Assembled & Tested $199.95
Kit Price Only $149.95

Features:

Lowpower 21L02, 450ns, Dip Switch Selectable addressing down to 256 Byte blocks, No wait states, fully buffered, battery back-up.

★ Plus the first 32K Static Memory Board Uses 2114 4K RAMS ★

16K	$549.00
24K	729.00
32K	945.00

DATA BOOKS

NSC Digital	$ 3.95
NSC Linear	4.95
NSC Linear A/N Vol. I	2.95
NSC Linear A/N Vol. II	2.95
NSC CMOS	2.95
NSC Audio	2.95
NSC Voltage Reg	2.95
NSC Memory	3.95
Intel Dete Book	4.95
NSC Special Functions	3.25
NSC Mos/LSI	4.25
NSC Transistors	2.95
Iasis Microprocessor Handbook	7.95
Iasis Complete up Learning Course	99.50
Intel MCS80 Menual	4.95
Intel MCS40 Manual	4.95
AMI MOS Catalog	2.95
Raytheon Linear	2.95
Reytheon Quads/Duels	1.95
New GI mos	4.95
Osborne Introto Microputer Vol. I	7.50
Osborne Introto Microputers Vol. II	12.50
Osborne 8080 Programming for Logic Design	7.50
Osborne 6800 Programming	7.50

SPECIALS

MCT2	$ .89
2N3638A	5/1.00
2N3640	5/1.00
2N2369A	5/1.00
2N3692	5/1.00
MSPS4318	5/1.00
MPS5401	5/1.00
MPS6516	5/1.00
MPS6544	5/1.00
MPS3568	5/1.00
MPS6522	5/1.00
MPS404A	5/1.00
2N5462	5/1.00
MPS3642	5/1.00
MPS6571	5/1.00
MPS3693	5/1.00
MPS6530	5/1.00
MPS3646	5/1.00
7525J	.50
7524J	.60
74S258	.60
7489	1.49
74145	.60
74H55	.19
715DC	1.49
74199	1.00

Plus huge backup inventory with special pricing for VOLUME requirements. Call for a quotation

BRIDGE SPECIAL

MDA 952-2	$1.49
MDA 962-3	1.49
KBPC2504	1.95

FREE 1977 CATALOG 56 pages
Send .25 for postage

COMPUTER KITS & SYSTEMS

MEK6800D2 Kit	$235.00
KIM1 6502	245.00
EVK99 6800 Kit	133.00
Intercept Jr. 6100 Kit	281.00
Iasis 8080 Computer Book	499.00
NSC SC/MP Kit	99.00
NSC Keyboard Kit	95.00
Low Cost S-100-BUS with 8 slots & Power Supply	159.95
8K Ram Kit (Logos I)	149.95
Byteuser 8K Eprom Kit	64.95
ZPU Kit (Z80)	295.00
TDL ZPU Kit (Z80)	269.00
Tarbell cassette I/O	115.00
S100 8 slot Motherboard with connectors (expandable)	69.95
S100 Extender Board	15.95
S100 Proto Board	27.95
Vector 8800 Proto Board	19.95
IMSAI 8080 w/22 slot	699.00
Cromemco Z-2	595.00
TDL XITAN alpha 1	769.00
Computalker	385.00
Hevristics Speechlab	299.00
SOROC IQ120 Terminal	995.00
8K Ram Board (Logos)	28.95
8K Eprom Board	24.95
S100 32K Ram Kit	945.00
Bytesaver Kit	145.00

All Shipments FCM or UPS. Orders under $100.00 add 5% handling and postage. Orders over $100.00 add 2.5% handling & postage. Mastercharge/Bank-americard/COD accepted w/25% deposit. California Residents add 6% tax. Foreign Orders add 8% handling. All parts prime factory tested guaranteed. Same day shipment. Add 25 cents for Data.

LOWEST PRICES YET

2708 Eprom 450 HS	$19.95
2716 16K Eprom	36.95
MK4116 (upD416) 16K Ram	34.95
Intel 3404 6 bit latch	2.95
2114 4K Static	12.95
upD410 4K Static	11.95
4200A 4K Static	12.95
1488 or 1489	1.25
8T97	1.25
8130	2.45
8131	2.45
MH0026CN	2.50
8833	1.85
8835	1.85
74367/74368	6/5.00
WD 4060 4K Ram	3.25
1702A-6	10/32.95
75452	10/2.50
8T26	2.25
H RAM 64 bit Ram	.75
78L05 Volt. Reg.	only .59
LM318H	1.00

BYTEUSER 8K EPROM WITH POWER ON JUMP/RESET JUMP USES 2708

Byteuser Kit	only $ 64.95
PCB only	24.95
8K Eprom ((8)(2708))	125.00

SPECIALTY CHIPS

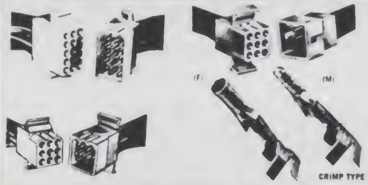
MPD371 Mag. Tape Control	$49.95
AYS 3550 4¾ DVM	24.95
MK5007/MK5009 Counter	6.95
AYS 3507 DVM	12.95
ICM7208IPI Ctr./Disp/Driver	16.95
ICM7045IPI Stop Watch	18.95
G115M, Intersil Analog Switch	1.50

Retail pricing may vary from Mail Order Pricing.

ADVANCED COMPUTER PRODUCTS

P. O. BOX 17329 K
Irvine, California 92713
New Phone (714) 558-8813

★ NOW Our First Store Open
1310 B East Edinger
Santa Ana, CA 92705 ★



NYLON CONNECTORS

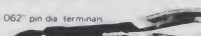
Mfg by Molex

Complete Connectors Per Pkg.	Type No.	Class	Description	Price Ea. Pkg.
5	1625-1PRT	Min.(.062")	1 Circuit	$1.75
3	1625-2PRT	"	2 Circuit	1.90
3	1625-3PRT	"	3 Circuit	2.10
2	1625-4PRT	"	4 Circuit	2.10
2	1625-5PRT	"	5 Circuit	2.20
2	1625-6PRT	"	6 Circuit	2.35
1	1649-8PRT	"	8 Circuit	1.55
1	1625-9PRT	"	9 Circuit	1.75
1	1625-12PRT	"	12 Circuit	1.90
1	1625-15PRT	"	15 Circuit	2.30
1	1625-24PRT	"	24 Circuit	3.25
1	1772-36PRT	"	36 Circuit	4.55
5	1619PRT	Std.(.093")	1 Circuit	1.75
3	1545PRT	"	2 Circuit	1.90
3	1396PRT	"	3 Circuit	2.10
2	1490PRT	"	4 Circuit	2.10
2	1653PRT	"	5 Circuit	2.20
2	1261PRT	"	6 Circuit	2.35
1	1292PRT	"	9 Circuit	1.80
1	1360PRT	"	12 Circuit	1.90
1	1375PRT	"	15 Circuit	2.45

Prototype hand tools combine efficiency with economy Ideal for prototype or limited production runs

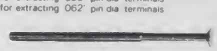
HT 1919 for 093" pin dia terminals $8 95 each

HT-1921 for 062" pin dia terminals $8 95 each



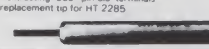
Econo-Extractor removes terminal from nylon connector housing with smoothness and ease

HT 2054 for extracting 093" pin dia terminals $2 25 each
HT 2023 for extracting 062' pin dia terminals $2 25 each



Deluxe ejector tools, spring loaded for simple efficient removal of terminal from nylon connector housing-extracts either male or female terminals of same pin diameter

HT 2038 for extracting 093" pin dia terminals $6 70 each
HT-1010-282 Replacement tip for HT 2038 $2 50 each
HT 2285 for extracting 062" pin dia terminals $6 70 each
HT 1672 3 replacement tip for HT 2285 $2 50 each

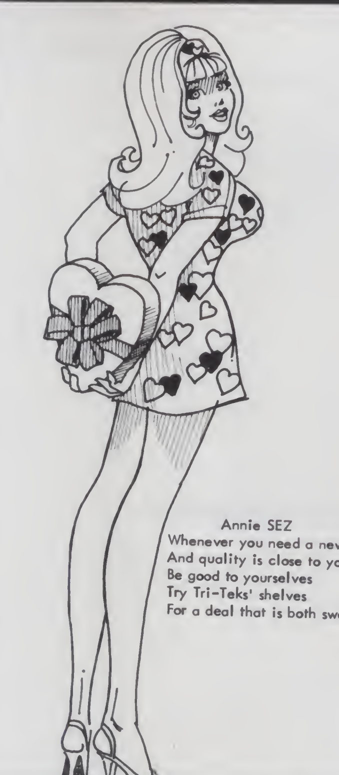


INTEGRATED CIRCUITS

555 Timer 8 pin mini-DIP	.49
741 Compensated OP-Amp 8 pin DIP	.37
LM 1889N RF Video Modulator	7.45
CA3130 Bipolar/Mos-FET Op Amp	1.19
CA3140 MOS-FET Op Amp, Bi-polar out	.99
LM3909 Lo Voltage Led Pulser	1.50
LM3911 Temp Control CHIP	1.50
Signetics 2504TA 1024 bit S.R. memory (1404A)..	$1.75
MCM 6571P Character Generator	9.95
MCM6571AP Character Generator	9.95
MCM6575P Character generator	$14.88
50240 Top octave generator	$14.95
LH0070-1H Precision (.3%) 10V Reference Amp	5.35
LHOO70-2H Extra Precise (.05%) 10V Reference Amp	10.55
LM399H Temp Stabilized Zener	5.95
AF100-1CJ Active Filter, State Variable	7.50
LM2907N Tachometer F/V Converter	2.65
LM1812N Ultra-sonic Transceiver	9.15
LM1815 Adaptive Sense Amp for Tachometer	5.73
TL170 TO-92 Hall effect switch w/spec sheets	1.25
MC14409P Telephone Rotary Pulser	10.98
MC14419P Touch Pad Converter for 14409	4.25
MC14411P Baud Rate Generator	11.98
MC14412VP CMOS Modem Chip	16.95
MM57109N Number Cruncher Micro	18.95
74C915 7 Segment to BCD Converter	2.99
74C922 16 Key Keyboard Encoder	6.35
74C923 20 key Keyboard Encoder	6.45
74C925 4 Decade Counter w/latches	12.00
74C926 4 Decade Counter *w/carry*	12.00
74C935-1 3½ Digit DVM CMOS Chip	16.98
9601 Retriggerable One shot	.50
MC4015P Hi Speed quad "D" low power TTL	$1.00

New UHF Prescaler IC. Nationals' DS8629 is a fixed ratio, divide by 100 counter combining ECL and lo-power Schottky. Single end or differential mode. Operates from D.C. to typically 160 MHz (135 Guaranteed minimum) TTL compatible, single supply, positive or negative edge trigger. 100mV to 1V input.

DS8629N $6.14
Specs... .40



Annie SEZ
Whenever you need a new part
And quality is close to your heart
Be good to yourselves
Try Tri-Teks' shelves
For a deal that is both sweet and smart

ZIF-14	$3.20	ZIF-22	4.50
ZIF-16	3.49	ZIF-24	4.85
ZIF-18	3.95	ZIF-40	7.19

(Last two digits of part number show number of pins per socket)

Unique Features of the Welcon Zero Insertion Force Dual-In-Line Sockets . . .

- **Capability to accept devices with up to .070" lead spread variation.**
- **Accepts lead lengths as short as .075".**
- **Sturdy terminals that fit .025" diameter printed circuit board holes.**
- **"Zero" insertion, positive wiping contacts.**
- **"ZIF" eliminates costly damage to fragile LSI devices**

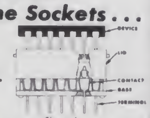


Figure 1
Device before insertion

Why struggle to insert DIP's into sockets when you can do it the easy way . . . the Welcon way . . . with a new design concept that provides "zero insertion force" and positive wiping.

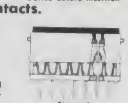


Figure 2

The "604" series of sockets, with all of its positive features and higher reliability, is small in size. High density packaging can be achieved for burn-in and production.

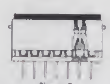
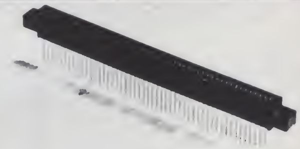


Figure 3

OPERATION: Lid is moved to up position stop (Fig. 2). This cams contacts into "open" position. DIP device can then be dropped into open contacts. In Fig. 3 DIP is pushed downward and contacts begin to close. When tip of device lead is past contact point (Fig. 4), contacts close and wipe on lead for remaining distance.

Figure 4



S-100 BUS CONNECTORS (IMSAI TYPE)

2x50x.125 x .25 Rows.

Gold, Solder tail for Mother boards $4.50,4/$17.00
Tin-Nickel, (NASGLO) Solder tail $3.75,4/$14.00
Gold, wire-wrap................ $4.50,4/$17.00
Tin-Nickel, (NASGLO) wire-wrap $3.75,4/$14.00

I.C. SOCKETS

Lo Profile Tin Solder Tail Dip Sockets

8 pin	10/$1.50	100/$14.00	1000/$120.00
14 pin	10/$1.70	100/$16.00	1000/$140.00
16 pin	10/$1.90	100/$18.00	1000/$160.00

VOLTAGE REGULATORS

7805-06-08-12-15-24 TO220	95¢	5/$4.50
7905-06-08-12-15-24 TO 220	95¢	5/$4.50
78LO5A-12-15 4% 100 mA TO-92 Plastic		50¢
78HO5KC 5V 5A TO-3		9.15
78H12KC 12V 5A TO-3		9.15
78H15KC 15V 5A TO-3		9.15
Lm317K 1.5A *Adjustable* TO-3		4.99
Lm317T 1.5A *Adjustable* TO-220		3.99
Lm317MP .5A *Adjustable* TO-202		13.95
TL430C Adjustable Zener-*Think About It*		1.50
TL497C Switching Reg. & Inductor		9.50
RCA CA 3085 100 mA *Adjustable*		.60

Adjustable 3A Regulator. LM350 is an adjustable 3-terminal regulator capable of supplying in EXCESS of 3 Amps over a 1.2V to 33V range. Standard TO-3 package for easy mounting. Full protection from thermal, current and safe area overload. Completely floating and can be used to regulate at any voltage level as long as input-output difference is less than 35Volts. 0.1% regulation!

LM350K.................................. $10.95
Specs... .80

DIODES AND BRIDGES

IN4003 200 V 1 amp		12/$1.00
IN4004 400 V 1 amp		10/$1.00
IN4148 Hi Speed Signal	15/$1.00	100/$5.00
D-600 115 V, 100 mA Hi Speed Signal		20/$1.00
D2131 200 V, 25A Stud		85¢
D2135 400 V, 25A Stud		1.00
D2138 600 V, 25A Stud		1.50
D3289R 200 V, 160A Stud Anode		5.85
D3909-4 50 V, 45A Fast Recovery		2.00
IN4732A-47A 1W 5% Zeners		4/$1.00
13 Assorted *Brand New* Zener Diodes		1.00
50V 3 amp Epoxy Bridge		79¢
200V 30 amp Bridge		2.00
600V 4 amp Epoxy Bridge		1.49
600V 3 amp Stud Bridge		.89
SI-2 200V, 1.5A Gold Leads		15/$1.00
D1A-0030 30V DIAC		10/$1.00

MISCELLANEOUS

RG-174 Miniature 50Ω coax	50'/4.25
WSU-30 Wire Wrap/unwrap tool	5.95
WSU-30M Modified Wrap/unwrap tool	6.95
BW-630 Battery Operated Wrap Tool	34.95
—Free Wire with any Wrap Tool—	
Miniature Square .05/100V Monolithic Cap	10/2.00
2N4036 90V, 1A PNP Silicon TO-5	.50
2N6101 80V, 10A NPN HI GAIN TO-220	.50
SE7005 250V, ½A NPN Silicon TO-5 W/Flange	.60
6.3 VCT, 1.2A Transformer F41X	2.49
12V, 1A Transformer with 6' Power Cord	2.88

Fairchild Linear Data Book. Huge volume has 17 chapters of data, applications and definitions. Probably the biggest volume of its type today........$4.50ppd

Fairchild Bipolar Memory Data Book. ROMS, PROMS and RAMS are covered in this work with full engineering data. Price includes shipping!...........$3.75

Raytheon Linear Data Book. Covers many of the familiar as well as those for which Raytheon is the inovator. Contains valuable applications data as well as complete specifications.................$2.95ppd

Raytheon Micro Computer Components Book. Covers Raytheons' offerings in the field from the micro-puter to memories and adapters................$2.95ppd

TRI-TEK, INC.
7808 North 27th Avenue
Phoenix, Arizona 85021
(602) 995-9352

T1

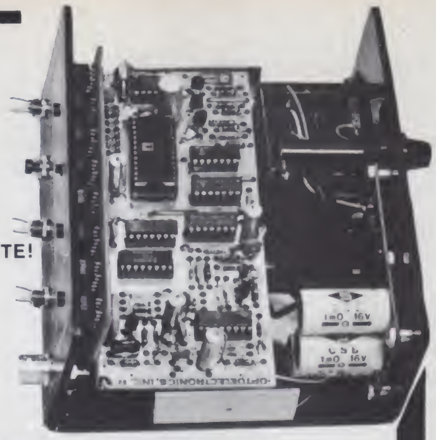
- Please give street address for UPS shipping when possible.
- C.O.D. NO parcel post C.O.D. UPS C.O.D. Add 85¢ to order.
- Any correspondence not connected with your order, please use seperate sheet and include SASE for reply.
- Orders less than $10 ($15 foreign) please add $1 handling.
- Prices are subject to change without notice.
- Any refunds will be by check, not credit vouchers.
- Terms. Check, money order, credit card. Net 30 days to rated firms, schools and government agencies.
- If we should be temporarily out of stock on an item, it will be placed on back order. If we cannot ship in 30 days, you will be notified of the expected shipping date and furnished with a postage paid card with which to cancel your order if desired.
- We pay surface shipping only in USA, Canada and Mexico.
- For premium shipping (first class, special handling, etc.) add extra. Excess will be refunded.
- Foreign orders (except Canada and Mexico) estimate and add shipping. Excess will be refunded.

Charge card telephone orders ($20 min.) will be accepted 9-5:30 P.M. except weekends. Telephone 995-9352. No collect calls please.

NEW LSI TECHNOLOGY

FREQUENCY COUNTER

TAKE ADVANTAGE OF THIS NEW STATE-OF-THE-ART COUNTER FEATURING THE MANY BENEFITS OF CUSTOM LSI CIRCUITRY. THIS NEW TECHNOLOGY APPROACH TO INSTRUMENTATION YIELDS ENHANCED PERFORMANCE, SMALLER PHYSICAL SIZE, DRASTICALLY REDUCED POWER CONSUMPTION [PORTABLE BATTERY OPERATION IS NOW PRACTICAL], DEPENDABILITY, EASY ASSEMBLY AND REVOLUTIONARY LOWER PRICING!



KIT #FC-50 C .... 60 MHZ COUNTER WITH CABINET & P.S...... $119.95 COMPLETE!
KIT #PSL-650 .... 650 MHZ PRESCALER [NOT SHOWN] 29.95
MODEL #FC-50WT....... 60 MHZ COUNTER WIRED, TESTED & CAL. 165.95
MODEL #FC-50/600 WT.. 600 MHZ COUNTER WIRED, TESTED & CAL. 199.95

SIZE:
3" High
6" Wide
5½" Deep

FEATURES AND SPECIFICATIONS:

DISPLAY: 8 RED LED DIGITS .4" CHARACTER HEIGHT
GATE TIMES: 1 SECOND AND 1/10 SECOND
PRESCALER WILL FIT INSIDE COUNTER CABINET
RESOLUTION: 1 HZ AT 1 SECOND, 10 HZ AT 1/10 SECOND.
FREQUENCY RANGE: 10 HZ TO 60 MHZ. [65 MHZ TYPICAL].
SENSITIVITY: 10 MV RMS TO 50 MHZ, 20 MV RMS TO 60 MHZ TYP.
INPUT IMPEDANCE: 1 MEGOHM AND 20 PF.
[DIODE PROTECTED INPUT FOR OVER VOLTAGE PROTECTION.]
ACCURACY: ± 1 PPM [± .0001%]; AFTER CALIBRATION TYPICAL.
STABILITY: WITHIN 1 PPM PER HOUR AFTER WARM UP [.001% XTAL]
IC PACKAGE COUNT: 8 [ALL SOCKETED]
INTERNAL POWER SUPPLY: 5 V DC REGULATED.
INPUT POWER REQUIRED: 8-12 VDC OR 115 VAC AT 50/60 HZ.
POWER CONSUMPTION: 4 WATTS

KIT #FC-50C IS COMPLETE WITH PREDRILLED CHASSIS ALL HARDWARE AND STEP-BY-STEP INSTRUCTIONS. WIRED & TESTED UNITS ARE CALIBRATED AND GUARANTEED.

PLEXIGLAS CABINETS

Great for Clocks or any LED Digital project Clear-Red Chassis serves as Bezel to increase contrast of digital displays.

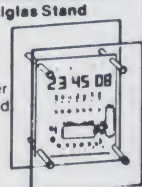
CABINET I 3"H,6¼"W,5½"D
CABINET II 2½"H,5"W,4"D

Black, White or Clear Cover $6.50 ea

RED OR GREY PLEXIGLAS FOR DIGITAL BEZELS 3"x6"x1/8" 95¢ ea 4/$3

SEE THE WORKS Clock Kit Clear Plexiglas Stand

- 6 Big .4" digits
- 12 or 24 hr. time
- 3 set switches
- Plug transformer
- all parts included



Plexiglas is Pre-cut & drilled

Kit #850-4 CP

Size: 6"H,4⅓"W,3"D

$23.50 ea 2/$45. Assembled $29.95

60 HZ. XTAL TIME BASE

Will enable Digital Clock Kits or Clock-Calendar Kits to operate from 12V DC. 1"x2" PC Board Power Req.: 5-15V (2.5 MA. TYP.) Easy 3 wire hookup Accuracy: ± 2PPM

#TB-1 (Adjustable)
Complete Kit $4.95
Wir & Cal $9.95

SPECIAL PRICING!

PRIME - HIGH SPEED RAM

21L02-3 400 NS

LOW POWER - FACTORY FRESH

1-24	$1.75 ea.	100-199	$1.45 ea.
25-99	1.60 ea.	200-999	1.39 ea.

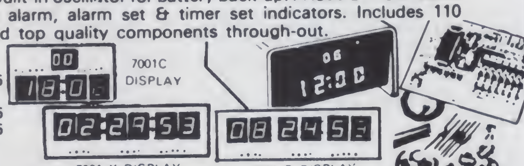
1000 AND OVER $1.29 ea.

6-DIGIT LED CLOCK CALENDAR KIT

DATE-TIME-SNOOZE ALARM & MORE... KIT 7001

FOR THE BUILDER THAT WANTS THE BEST. FEATURING 12 OR 24 HOUR TIME — 29-30-31 DAY CALENDAR. ALARM, SNOOZE AND AUX. TIMER CIRCUITS

Will alternate time (8 seconds) and date (2 seconds) or may be wired for time or date display only, with other functions on demand. Has built-in oscillator for battery back-up. A loud 24 hour alarm with a repeatable 10 minute snooze alarm, alarm set & timer set indicators. Includes 110 VAC/60Hz power pack with cord and top quality components through-out.



KIT - 7001B WITH 6 - .5" DIGITS $39.95
KIT - 7001C WITH 4 - .6" DIGITS & 2 - .3" DIGITS FOR SECONDS $42.95
KIT - 7001X WITH 6 - .6" DIGITS $45.95

KITS ARE COMPLETE (LESS CABINET)
ALL 7001 KITS FIT CABINET I AND ACCEPT QUARTZ CRYSTAL TIME BASE KIT # TB-1

JUMBO DIGIT CLOCK

A complete Kit (less Cabinet) featuring: six .5" digits, MM5314 IC 12/24 Hr. time, PC Boards, Transformer, Line Cord, Switches and all Parts. Ideal Fit in Cabinet II

Kit #5314-5 $19.95 2/$38.

JUMBO DIGIT CONVERSTION KIT $9.95 ea.

Convert small digit LED clock to large .5" displays. Kit includes 6 - LED's, Multiplex PC Board & Hook up info.
Kit #JD-1CC For Common Cathode
Kit #JD-1CA For Common Anode

PRINTED CIRCUIT BOARDS for CT-7001 Kits sold separately with assembly info. PC Boards are drilled Fiberglass, solder plated and screened with component layout.
Specify for 7001 B, C or X - $7.95

MOBILE LED CLOCK

12/24 HR .4" DIGITS!

MODEL #2001

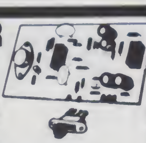
12 VOLT AC or DC POWERED

- 6 JUMBO .4" RED LED'S BEHIND RED FILTER LENS WITH CHROME RIM
- SET TIME FROM FRONT VIA HIDDEN SWITCHES • 12/24-Hr. TIME FORMAT
- STYLISH CHARCOAL GRAY CASE OF MOLDED HIGH TEMP. PLASTIC
- BRIDGE POWER INPUT CIRCUITRY — TWO WIRE NO POLARITY HOOK-UP
- OPTIONAL CONNECTION TO BLANK DISPLAY [Use When Key Off in Car, Etc.]
- TOP QUALITY PC BOARDS & COMPONENTS - INSTRUCTIONS.
- MOUNTING BRACKET INCLUDED

KIT #2001 COMPLETE KIT $27.95 EA. 3 OR MORE $25.95 ea. 115 VAC Power Pack #AC-1 $2.50 EA.

ASSEMBLED UNITS WIRED & TESTED ORDER #2001 WT [LESS 9V. BATTERY] Wired for 12-Hr. Op. if not otherwise specified. $37.95 EA 3 OR MORE $35.95 ea.

AUTO BURGLAR ALARM KIT



AN EASY TO ASSEMBLE AND EASY TO INSTALL ALARM PROVIDING MANY FEATURES NOT NORMALLY FOUND. KEYLESS ALARM HAS PROVISION FOR POS & GROUNDING SWITCHES OR SENSORS. WILL PULSE HORN RELAY AT 1HZ RATE OR DRIVE SIREN. KIT PROVIDES PROGRAMMABLE TIME DELAYS FOR EXIT, ENTRY & ALARM PERIOD. UNIT MOUNTS UNDER DASH. REMOTE SWITCH CAN BE MOUNTED WHERE DESIRED. CMOS RELIABILITY RESISTS FALSE ALARMS & PROVIDES FOR ULTRA DEPENDABLE ALARM. DO NOT BE FOOLED BY LOW PRICES! THIS IS A TOP QUALITY COMPLETE KIT WITH ALL PARTS INCLUDING DETAILED DRAWINGS AND INSTRUCTIONS OR AVAILABLE WIRED AND TESTED

KIT #ALR-1 $9.95
#ALR-1WT WIRED & TESTED $19.95

VARIABLE REGULATED 1 AMP POWER SUPPLY KIT

- VARIABLE FROM 4 to 14V
- SHORT CIRCUIT PROOF
- 723 IC REGULATOR
- 2N3055 PASS TRANSISTOR
- CURRENT LIMITING AT 1 Amp

KIT IS COMPLETE INCLUDING DRILLED & SOLDER PLATED FIBERGLASS PC BOARD AND ALL PARTS (Less TRANSFORMER) KIT #PS-01 $8.95

TRANSFORMER 24V CT will provide 300MA at 12V and 1 Amp at 5V. $3.50

ORDER BY PHONE OR MAIL
COD ORDERS WELCOME

OPTOELECTRONICS, INC.

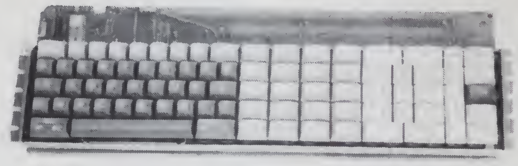
BOX 219 HOLLYWOOD, FLA. 33022
PHONE [305] 921-2056 / 921-4425 O3

ORDERS TO USA & CANADA ADD 5% FOR SHIPPING, HANDLING & INSURANCE. ALL OTHERS ADD 10%. ADDITIONAL $1.00 CHARGE FOR ORDERS UNDER $15.00 - COD FEE $1.00. FLA. RES. ADD 4% STATE TAX.

master charge
BANKAMERICARD

FANTASTIC KEYBOARDS

5567N

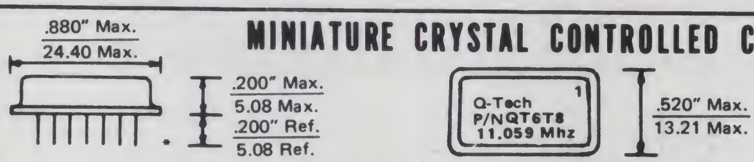


5566N

We keep coming up with fabulous keyboards. Unfortunately, they show up in limited quantities. Shown is STOCK NO. 5567N, a keyboard originally made for a bank computer. It has a total of 82 keys, including all alpha numeric functions. The keys are all HALL EFFECT, and the keyboard is ASCII encoded. Made by MICRO-SWITCH, (HONEYWELL), factory wrapped. On some boards, the extra keys are blank, and some have banking functions on them.
STOCK NO. 5566N, is similar to above, but has 78 keys. All keys are HALL EFFECT, and board is ASCII encoded.
Due to limited number of boards, and in order to avoid disappointment, we can only accept phone orders for these keyboards. Phone us at 617-388-4705

STOCK NO. 5567N 82 Key Keyboard $64.50 STOCK NO. 5566N 78 Key Keyboard $64.50

MINIATURE CRYSTAL CONTROLLED CLOCK OSCILLATORS



This complete crystal oscillator fits in a standard 14 pin DIP socket. It is TTL compatible, and will drive 10 TTL loads.

Crystal is .0005%. Output "High" is 2.8 volts Min. and "Low" is .4 volts Max. Vcc is 5 volts. An ususual state of the art device to provide your computer with a stable, reliable clock source. This device sells for $52.00 in 1 piece quantity. Operating frequency is 11.059 MHz.

STOCK NO. 1048K 11.059 Crystal Oscillator $11.95 2/22.00

INTEL RANDOM ACESS MEMORY (RAM)

The famous 2102 RAM. 1024 bits, fully decoded. First line INTEL, branded. Not rejects or retests.

STOCK NO. 3156K INTEL p2102 RAM

$2.00 ea. 5/9.00

A/D and DA CONVERTERS

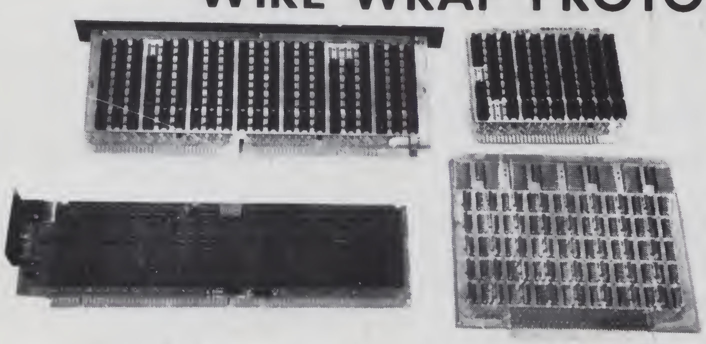
DATEL Model 898B is an 8 bit A to D converter. Input range 0 to 10 V, or ± 5 volts. Input imp. 4.25K ohms. Power supply requirements ± 15 V. and +5 V. Coding, 8 binary bits. Present Mfg. price, $69.00
STOCK NO. 5556K A/D converter, Datel 898B $28.75 2/54.00
DATEL D/A converter. Model 298. 8 bit resolution. TTL compatible. 5 MHz update rate. Output 0-10 or ± 5 V. Factory price $39.00
STOCK NO. 5558K Model 298 A D/A converter $18.95 2/36.00

VIDEOCUBE— TV COMPUTER INTERFACE KIT

VIDEOCUBE is a self contained RF oscillator and modulator which allows easy interface with any video output device to a standard TV set. This device was described in August RADIO-ELECTRONICS. We provide a reprint of the article. Approved by FCC for radiation. Kit contains all parts as shown in the RADIO-ELECTRONICS article.

STOCK NO. 5500K $13.95 2/26.00

WIRE WRAP PROTOTYPE BOARDS



Wire wrap is the thing today, whether you are adding memory to your computer, building from scratch, Prototyping or designing new circuits, etc. We have 4 boards. 2 out of equipment, and have wire on the pins that must be removed, (easy with an OK wire wrap tool, and 2 virgin boards. Board 6558K has from 75 to 100 sockets, both 14 and 16 pin. Board 6559K has from 40 to 50 sockets, 14 & 16 pin. Board 6592K has 40 16 pin sockets, & 4 LSI 24 pin sockets and is gold plated. All pins are brought up to top of board for ease in wiring. Board 5561K has 87½ sockets, 28 16 pin sockets and a 4 pin socket. These boards are all heavily by-passed between Vcc and ground planes. Some of the boards in this number contain 4 LEDs, 2 red, a yellow and a green and a 4 position thumb wheel switch, with all leads brought out to wire wrap pins.
All boards advertised have Vcc and ground planes, and all have gold contacts for standard edge connectors.

STOCK NO. 6558K 75 to 100 sockets, 5¾" x 13¾", removed from equipment $18.75 ea. 2/35.00
STOCK NO. 6559K 40 to 50 sockets, 6" x 6½", removed from equipment $11.75 ea. 2/22.00
STOCK NO. 6592K 40 16 pin sockets, 4 LSI sockets, 6¾" x 8¾" Gold Pl. New $24.50 2/45.00
STOCK NO. 5561K 88½ sockets, 4½" x 14½" New. $29.50 ea. 2/55.00

DELTA ELECTRONICS
7 Oakland St.
Amesbury, Mass. 01913
(617) 388-4705 D13

master charge THE INTERBANK CARD — Your BANKAMERICARD welcome here

ATLANTA, GEORGIA
DELTA ELECTRONIC HOBBIES, 5151 Buford Hwy., Doraville, Atlanta, Ga. Retail only.

BOSTON, MASSACHUSETTS
DELTA ELECTRONICS WAREHOUSE OUTLET, 590 Commonwealth Ave., Boston, Ma. Retail only.

DIODES/ZENERS

1N914	100v	10mA	.05
1N4005	600v	1A	.08
1N4007	1000v	1A	.15
1N4148	75v	10mA	.05
1N753A	6.2v	z	.25
1N758A	10v	z	.25
1N759A	12v	z	.25
1N4733	5.1v	z	.25
1N5243	13v	z	.25
1N5244B	14v	z	.25
1N5245B	15v	z	.25

SOCKETS/BRIDGES

8-pin	pcb	.25	ww	.45
14-pin	pcb	.25	ww	.40
16-pin	pcb	.25	ww	.40
18-pin	pcb	.25	ww	.75
22-pin	pcb	.45	ww	1.25
24-pin	pcb	.35	ww	1.10
28-pin	pcb	.35	ww	1.45
40-pin	pcb	.50	ww	1.25

Molex pins	.01	To-3 Sockets	.45
2 Amp Bridge		100-prv	1.20
25 Amp Bridge		200-prv	1.95

TRANSISTORS, LEDS, etc.

2N2222	NPN	(Plastic .10)	.15
2N2907	PNP		.15
2N3906	PNP		.10
2N3054	NPN		.35
2N3055	NPN	15A 60v	.50
T1P125	PNP	Darlington	.35
LED Green, Red, Clear			.15
D.L.747	7 seg 5/8" high com-anode		1.95
XAN72	7 seg com-anode		1.50
FND 359	Red 7 seg com-cathode		1.25

C MOS

4000	.15
4001	.20
4002	.20
4004	3.95
4006	1.20
4007	.35
4008	.95
4009	.30
4010	.45
4011	.20
4012	.20
4013	.40
4014	1.10
4015	.95
4016	.35
4017	1.10
4018	1.10
4019	.60
4020	.85
4021	1.35
4022	.95
4023	.25
4024	.75
4025	.35
4026	1.95
4027	.50
4028	.95
4030	.35
4033	1.50
4034	2.45
4035	1.25
4040	1.35
4041	.69
4042	.95
4043	.95
4044	.95
4046	1.75
4049	.70
4050	.50
4066	.95
4069	.40
4071	.35
4081	.70
4082	.45

– T T L –

7400	.15
7401	.15
7402	.20
7403	.20
7404	.15
7405	.25
7406	.35
7407	.55
7408	.25
7409	.15
7410	.10
7411	.25
7412	.30
7413	.45
7414	1.10
7416	.25
7417	.40
7420	.15
7426	.30
7427	.45
7430	.15
7432	.30
7437	.35
7438	.35
7440	.25
7441	1.15
7442	.45
7443	.85
7444	.45
7445	.65
7446	.95
7447	.95
7448	.70
7450	.25
7451	.25
7453	.20
7454	.25
7460	.40
7470	.45
7472	.40

7473	.25
7474	.35
7475	.35
7476	.30
7480	.55
7481	.75
7483	.95
7485	.95
7486	.30
7489	1.35
7490	.55
7491	.95
7492	.95
7493	.40
7494	1.25
7495	.60
7496	.80
74100	1.85
74107	.35
74121	.35
74122	.55
74123	.55
74125	.45
74126	.35
74132	1.35
74141	1.00
74150	.85
74151	.75
74153	.95
74154	1.05
74156	.95
74157	.65
74161	.85
74163	.95
74164	.60
74165	1.50
74166	1.35
74175	.80

74176	1.25
74180	.85
74181	2.25
74182	.95
74190	1.75
74191	1.35
74192	1.65
74193	.85
74194	1.25
74195	.95
74196	1.25
74197	1.25
74198	2.35
74221	1.00
74367	.85
75108A	.35
75110	.35
75491	.50
75492	.50
74H00	.25
74H01	.25
74H04	.25
74H05	.25
74H08	.35
74H10	.35
74H11	.25
74H15	.30
74H20	.30
74H21	.25
74H22	.40
74H30	.25
74H40	.25
74H50	.25
74H51	.25
74H52	.15
74H53J	.25
74H55	.25

74H72	.55
74H101	.75
74H103	.75
74H106	.95
74L00	.35
74L02	.35
74L03	.30
74L04	.35
74L10	.35
74L20	.35
74L30	.45
74L47	1.95
74L51	.45
74L55	.65
74L72	.45
74L73	.40
74L74	.45
74L75	.55
74L93	.55
74L123	.55
74S00	.55
74S02	.55
74S03	.30
74S04	.35
74S05	.35
74S08	.35
74S10	.35
74S11	.35
74S20	.35
74S40	.25
74S50	.25
74S51	.45
74S64	.25
74S74	.40
74S112	.90
74S114	1.30

74S133	.45
74S140	.75
74S151	.35
74S153	.35
74S157	.80
74S158	.35
74S194	1.05
74S257 (8123)	.25
74LS00	.35
74LS01	.35
74LS02	.35
74LS04	.35
74LS05	.45
74LS08	.35
74LS09	.35
74LS10	.35
74LS11	.35
74LS20	.35
74LS21	.25
74LS22	.25
74LS32	.40
74LS37	.35
74LS40	.45
74LS42	1.10
74LS51	.50
74LS74	.65
74LS86	.65
74LS90	.95
74LS93	.95
74LS107	.85
74LS123	1.00
74LS151	.95
74LS153	1.20
74LS157	.85
74LS164	1.90
74LS367	.85
74LS368	.85

9000 SERIES

9301	.85
9309	.35
9322	.85
95H03	.55
9601	.75
9602	.50

MEMORY CLOCKS

74S188 (8223)	3.00
1702A	6.95
MM5314	3.00
MM5316	3.50
2102-1	1.75
2102L-1	1.95
TR 1602B/ TMS 6011	6.95
8080AD	15.00
8T13	1.50
8T23	1.50
8T24	2.00
2107B-4	4.95

LINEARS, REGULATORS, etc.

8266	.35
MCT2	.95
8038	3.95
LM201	.75
LM301	.25
LM308 (Mini)	.75
LM309H	.65
LM309K (340K-5)	.85
LM310	1.15
LM311D (Mini)	.75
LM318 (Mini)	.65

LM320K5 (7905)	1.65
LM320K12	1.65
LM320T5	1.65
LM320T12	1.65
LM320T15	1.65
LM339	.95
7805 (340T5)	.95
LM340T12	1.00
LM340T15	1.00
LM340T18	1.00

LM340T24	.95
LM340K12	2.15
LM340K15	1.25
LM340K18	1.25
LM340K24	.95
LM373	2.95
LM380	.95
LM709 (8,14 PIN)	.25
LM711	.45

LM723	.50
LM725	1.75
LM739	1.50
LM741 (8-14)	.25
LM747	1.10
LM1307	1.25
LM1458	.95
LM3900	.50
LM75451	.65
NE555	.50
NE556	.95
NE565	.95
NE566	1.75
NE567	1.35

INTEGRATED CIRCUITS UNLIMITED

7889 Clairemont Mesa Boulevard, San Diego, California 92111
(714) 278-4394 *(Calif. Res.)*

All orders shipped prepaid — No minimum
Open accounts invited — COD orders accepted

Discounts available at OEM Quantities — California Residents add 6% Sales Tax
All IC's Prime/Guaranteed. All orders shipped same day received.

24 Hour Toll Free Phone 1-800-854-2211 — **MasterCharge / BankAmericard / AE**

SPECIAL DISCOUNTS

Total Order	Deduct
$35 - $99	5%
$100 - $300	10%
$301 - $1000	15%
$1000 - Up	20%

1977 kilobaud Binders

Keep your Kilobauds from being lost or damaged in these handsomely appointed binders with rich blue covers and gold lettering. Each binder holds 12 issues, making an EXCELLENT REFERENCE HANDBOOK. Several binders form a quality library you can be proud of.

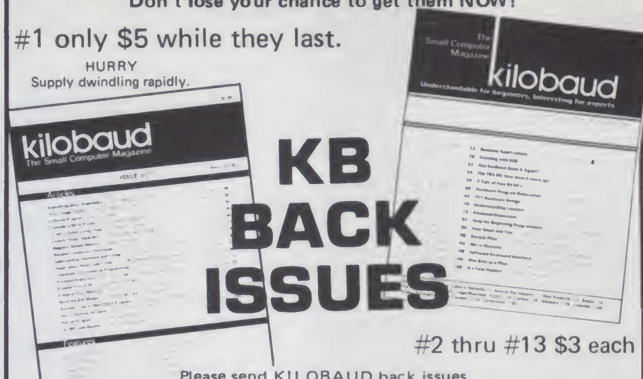
Order yours today . . . only $6.00 each . . . $5.00 each additional.

Use the order card in the back of this magazine or itemize your order on a separate piece of paper and mail to:
Kilobaud Binders Peterborough NH 03458
Be sure to include check or detailed credit card information
Add $1.00 shipping & handling charge for each order.

KB/2/78

Did you miss out on the first issue of Kilobaud?
Don't lose your chance to get them NOW!

#1 only $5 while they last.
HURRY
Supply dwindling rapidly.



#2 thru #13 $3 each

Please send KILOBAUD back issues.

___ issues JAN 77 $5	___ issues MAY 77 $3	___ issues SEPT 77 $3
___ issues FEB 77 $3	___ issues JUNE 77 $3	___ issues OCT 77 $3
___ issues MAR 77 $3	___ issues JUL 77 $3	___ issues NOV 77 $3
___ issues APR 77 $3	___ issues AUG 77 $3	___ issues DEC 77 $3
	___ TOTAL	___ issues JAN 78 $3

☐ BankAmericard ☐ Master Charge ☐ American Express

Card # ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐

Expiration date__________

Signature______________________________

Name______________________________

Address______________________________

City__________________ State______ Zip______

*Add $1.00 shipping & handling charge for each order

Have credit card # handy and call
TOLL FREE (800) 258-5473
KILOBAUD ● PETERBOROUGH NH 03458

KB/2/78

IT'S EASY AND FUN

TO BUILD YOUR OWN TEST EQUIPMENT WITH ICs

ICs have greatly simplified even the most sophisticated pieces of test equipment, making them fun to build. You can save thousands of dollars by building your own equipment and have a test lab which would make a university jealous.

A brand new book, IC TEST EQUIPMENT, has construction projects for making 37 pieces of test equipment. Square wave generator, pulse generator, timer, audio synthesizer, AFSK generator, sync generator, counters, capacity meter, etc.

Order this book today and get started building your own lab.

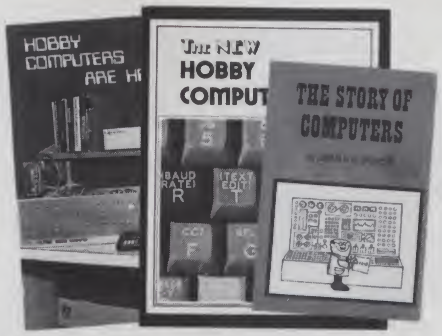
IC TEST EQUIPMENT $4.95*

Use the order card in the back of the magazine or itemize your order on a separate piece of paper and mail to: KILOBAUD BOOK NOOK, Peterborough NH 03458 or phone toll free 800-258-5473

Be sure to include check or detailed credit information. *Add $1 shipping and handling charge for each order.

kb book nook

beginner's•introductory



●**THE STORY OF COMPUTERS** by Donald D. Spencer is to computer books what *Dick and Jane* is to novels . . . extremely elementary, gives the non-computerist a fair idea of what the hobbyist is talking about when he speaks computer lingo. Attempts to explain what computers are and can do to a spouse, child or any un-electronics-minded friend. $4.95.*

●**HOBBY COMPUTERS ARE HERE** If you (or a friend) want to come up to speed on how computers work . . . hardware and software . . . this is an excellent book. It starts with the fundamentals and explains the circuits, the basics of programming, along with a couple of TVT construction projects, ASCII-Baudot, etc. This book has the highest recommendations as a teaching aid for newcomers. $4.95.*

●**THE NEW HOBBY COMPUTERS!** This book takes it from where "Hobby Computers Are Here" leaves off, with chapters on Large Scale Integration, how to choose a microprocessor chip, an introduction to programming, low cost I/O for a computer, computer arithmetic, checking memory boards, a Baudot monitor/editor system, an audible logic probe for finding those tough problems, a ham's computer, a computer QSO machine . . . and much, much more! $4.95.*

●**HOME COMPUTERS: 2^{10} Questions & Answers** by Rich Didday. Two books aimed exclusively at the novice computer hobbyist/home computer user. Written in a rather unusual style which has a beginner asking questions which are answered by a person with a substantial background in computers and personal computing. The questions are just the kind beginners come up with . . . and the answers are presented in easy-to-understand terms (usually with a diagram to illustrate the point). Both the hardware and software aspects of home computing are covered from A to Z. An index in both books makes them ideal as reference material *for anyone*. Volume I: Hardware — $7.95*; Volume 2: Software — $6.95*.

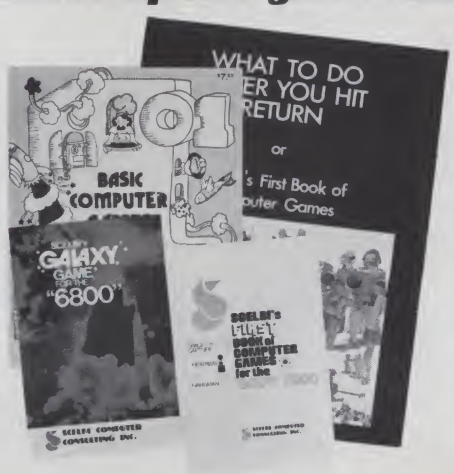
●**MICROPROCESSORS FROM CHIPS TO SYSTEMS** by Rodnay Zaks is a complete and detailed introduction to microprocessors and microcomputer systems. No preliminary knowledge of computers or microprocessors is required to read this book, although a basic engineering knowledge is naturally an advantage. Intended for all wishing to understand the concepts, techniques and components of microprocessors in a short time. $9.95.*

●**MICROCOMPUTER PRIMER** by Mitchell Waite and Michael Pardee. Describes basic computer theory, explains numbering systems, and introduces the reader to the world of programming. Describes the world of microcomputing in "real world" terminology. No better way of getting involved with the exciting new hobby of microcomputing. $7.95.*

●**INTRODUCTION TO MICROPROCESSORS** by Charles Rockwell of MICROLOG is an ideal reference for the individual desiring to understand the hardware aspects of microprocessor systems. Describes the hardware details of computer devices in terms the beginner can understand, instead of treating the micro chip as a "black box." General information about hardware systems is provided. Specific systems are not described and programming is only briefly discussed. $17.50 US and Canada, $20 elsewhere.*

●**AN INTRODUCTION TO MICROCOMPUTERS, VOLS. 1 AND 2** by Adam Osborne Associates, are references dealing with microcomputer architecture in general and specifically with details about most of the common chips. These books are not software-oriented, but are invaluable for the hobbyist who is into building his own interfaces and processors. Volume 1 is dedicated to general hardware theory related to micros, and Volume 2 discusses the practical details of each micro chip. (Detailed review in Kilobaud #2) Volume 1 — $7.50*; Volume II — $12.50.*

computer games



●**WHAT TO DO AFTER YOU HIT RETURN** PCC's first book of computer games . . . 48 different computer games you can play in BASIC . . . programs, descriptions, muchly illustrated. Lunar landing, Hammurabi, King, Civel 2, Qubic 5, Taxman, Star Trek, Crash, Market, etc. $8.00.*

●**SCELBI'S GALAXY GAME FOR THE "6800"** Here's a new twist in computer games by Robert Findley/Raymond Edwards. "Galaxy" pits the operator of a spaceship against alien craft, as well as such variables as speed, time, and ammunition. No two games are the same! $14.95.*

●**101 BASIC COMPUTER GAMES** Okay, so once you get your computer up and running in BASIC, then what? Then you need some programs in BASIC, that's what. This book has 101 games for you, from very simple to real buggers. You get the games, a description of the games, the listing to put in your computer and a sample run to show you how they work. Fun. Any one game will be worth more than the price of the book for the fun you and your family will have with it. $7.50.*

●**SCELBI'S FIRST BOOK OF COMPUTER GAMES** Need a game for your 8008 or 8080 microprocessor? Has three popular games, "Space Capture," "Hexpawn," and "Hangman." Complete flowcharts, logic description, program listing, and instructions are provided. A must for the game freak! $14.95.*

Use the order card in the back of this magazine or itemize your order on a separate piece of paper and mail to:
Kilobaud Book Department Peterborough NH 03458
Be sure to include check or detailed credit card information
* *Add $1.00 shipping & handling charge for each order.* Note: Prices subject to change on books not published by 73 Magazine.

software·programming

•DISCOVERING BASIC — A Problem Solving Approach by Robert E. Smith deals with progressively more complex problems which allow the reader to discover the vocabulary of BASIC language as he develops skill and confidence in putting it to work. Clear and concise explanations. Problems used cover a wide range of interests — insurance, geometry, puzzles, economics, etc. $6.85.*

•BASIC by Bob Albrecht, etc. Self-teaching guide to the computer language you will need to know for use with your microcomputer. This is one of the easiest ways to learn computer programming. $4.95.*

•A QUICK LOOK AT BASIC by Donald D. Spencer. A perfect reference for the beginning programmer. Assumes that the reader has no previous programming experience and is a self-teaching guide for the individual desiring to learn the fundamentals of BASIC. $4.95.*

•MY COMPUTER LIKES ME . . . WHEN I SPEAK BASIC An introduction to B/ SIC . . . simple enough for your kids. If you want to teach BASIC to anyone quick' , this book is the way to go. $2.00.*

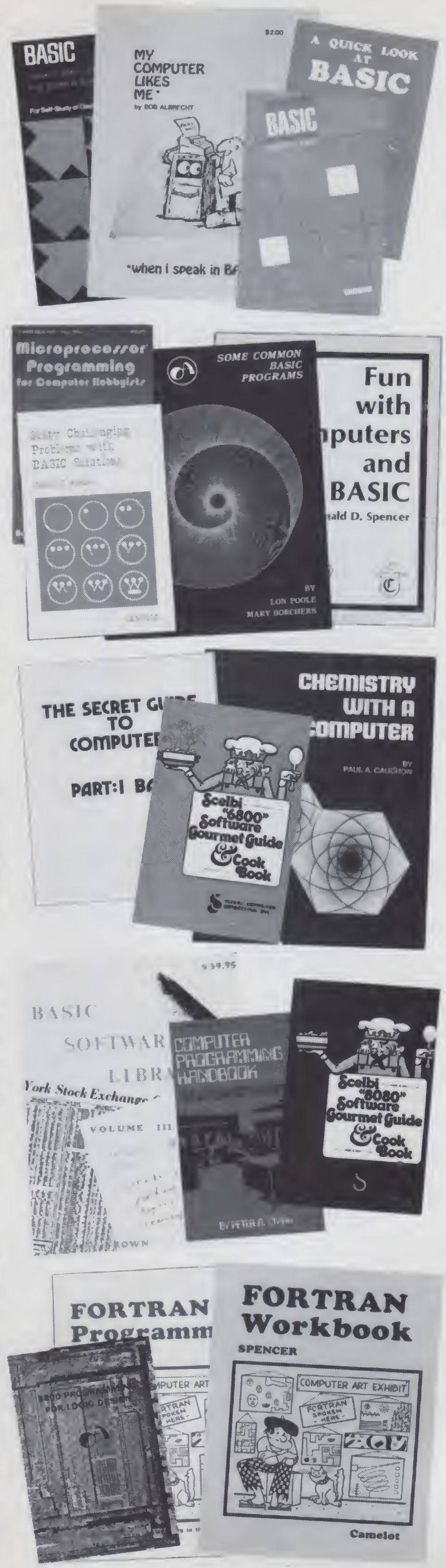
•FUN WITH COMPUTERS AND BASIC by Donald D. Spencer, contains an easy-to-understand explanation of the BASIC Programming Language and is intended for persons who have had no previous exposure to computer programming. Over half the book is devoted to problems using games, puzzles, and mathematical recreations. A superior book for self-teaching and learning computer programming. $6.95.*

• SIXTY CHALLENGING PROBLEMS WITH BASIC SOLUTIONS by Donald Spencer, provides the serious student of BASIC programming with interesting problems and solutions. No knowledge of math above algebra required. Incudes a number of game programs, as well as programs for financial interest, conversions and numeric manipulations. $6.95.*

•THE SECRET GUIDE TO COMPUTERS Parts 1, 2, and 3 by Russ Walter. **Part One** describes computers in general, and after reading for ten minutes you will be writing simple BASIC programs! **Part Two** discusses computer applications. It's one thing to master the syntax of a language such as BASIC and another to solve problems using the new tool. **Part Three** describes programming languages. Ever heard of APL and QLISP? BASIC is not the only language used to program computers. 7th Edition. Part I - $2.75*; Part II — $2.50*; Part III — $3.50.*

•SOME COMMON BASIC PROGRAMS published by Adam Osborne & Associates, Inc. Perfect for non-technical computerists requiring ready-to-use programs. Business programs, plus miscellaneous programs. Invaluable for the user who is not an experienced programmer. All will operate in the standalone mode. $7.50 paperback.*

•Scientific Research Instruments' BASIC SOFTWARE LIBRARY is a complete do-it-yourself kit. Written in everybody's BASIC immediately executable in ANY computer with at least 4K, no other peripherals needed. Vol. I contains business and recreational programs and is 300 pages. Vol. II is 260 pages and contains math, engineering, statistics and plotting programs. Vol. III contains money managing, advanced business programs such as billing, A/R, inventory, payroll, etc. Vol. IV contains general purpose programs like loans, rates, retirement, plus games: Poker, Enterprise (take charge while Capt. Kirk is away), Football and more! Vol. V is filled with experimenter's programs including games, pictures and misc. problems like "logic." Vols. I & II $24.95,* Vol. III $39.95,* Vol. IV & V $9.95 each.*



•MICROPROCESSOR PROGRAMMING for Computer Hobbyists by Neill Graham is for the hobbyist interested in intermediate and advanced techniques of programming and data structuring. Written to take up where the computer manufacturers' instruction manuals and the introductory programming language texts leave off. $8.95.*

• COMPUTER PROGRAMMING HANDBOOK by Peter Stark. A complete guide to computer programming and data processing. Includes many worked out examples and history of computers. $8.95.*

• 8080 PROGRAMMING FOR LOGIC DESIGN Ideal reference for an in-depth understanding of the 8080 processor. Application-oriented and the 8080 is discussed in light of replacing conventional, hard-wired logic. Practical design considerations are provided for the implementation of an 8080-based control system. $7.50.*

•8080 SOFTWARE GOURMET GUIDE AND COOKBOOK If you have been spending too much time developing simple routines for your 8080, try this new book by Scelbi Computing and Robert Findley. Describes sorting, searching, and many other routines for the 8080 user. $9.95.*

• 6800 PROGRAMMING FOR LOGIC DESIGN Oriented toward the industrial user, this book describes the process by which conventional logic can be replaced by a 6800 microprocessor. Provides practical information that allows an experimenter to design a complete micro control system for the "ground up." $7.50.*

•6800 SOFTWARE GOURMET GUIDE & COOKBOOK If you have been spending too much time developing routines for your 6800 microprocessor, try the new book by Scelbi Computing and Robert Findley. Describes sorting, searching, and many other routines for the 6800 user. $9.95.*

•FORTRAN PROGRAMMING by Donald Spencer. FORTRAN was designed for complex numeric calculations; and possesses extended I/O capability. It is easily learned, as it is an English-like computer language. $7.50.*

• FORTRAN WORKBOOK by Donald Spencer. Provides practical examples and problems to solve. Flowcharting is also discussed. Almost all micros support BASIC — it won't be long before FORTRAN is commonplace. $3.95.*

•CHEMISTRY WITH A COMPUTER by Paul A. Cauchon, contains a collection of tutorial, simulation and problem-generation computer programs. Usable with almost any chemistry course in the high school or college level. $9.95.*

Use the order card in the back of this magazine or itemize your order on a separate piece of paper and mail to:
Kilobaud Book Department Peterborough NH 03458
Be sure to include check or detailed credit card information

** Add $1.00 shipping & handling charge for each order.* Note: Prices subject to change on books not published by 73 Magazine.

Kb book nook

Kb book nook

kb book nook

hardware

● **MICROPROCESSOR INTERFACING TECHNIQUES** by Austin Lesea & Rodnay Zaks will teach you how to interconnect a complete system and interface it to all the usual peripherals. It covers hardware and software skills and techniques, including the use and design of model buses such as the IEEE 488 or $100. $9.95.*

● **TTL COOKBOOK** by Donald Lancaster. Explains what TTL is, how it works, and how to use it. Discusses practical applications, such as a digital counter and display system, events counter, electronic stopwatch, digital voltmeter, and a digital tachometer. $8.95.*

● **CMOS COOKBOOK** by Don Lancaster. Details the application of CMOS, the low power logic family suitable for most applications presently dominated by TTL. Required reading for every serious digital experimenter! $9.95.*

● **TVT COOKBOOK** by Don Lancaster, describes the use of a standard television receiver as a microprocessor CRT terminal. Explains and describes character generation, cursor control and interface information in typical, easy-to-understand Lancaster style. $9.95.*

● **BUILD-IT BOOK OF DIGITAL ELECTRONIC TIMEPIECES** by Robert Haviland is a data-packed guide to building every timekeeping device you can imagine: rugged shipboard clocks, second-splitting digital IC chronometers, decorator digital clocks, a precision timer, a frequency-period meter, a tide and moon clock, an automatic alarm setter, etc. Including full-size printed circuit board layouts. $6.95.*

● **THE "COMPULATOR" BOOK – Building Super Calculators & Minicomputer Hardware with Calculator Chips** by R. P. Haviland, provides ideas, design info and printed circuit boards for calculator chip projects like "training" an inexpensive "garden variety" calculator chip to dial a telephone, measure time, tie in with a Teletype to create a virtually infinite memory system, and countless other functions. $7.95.*

● **VOL I COMPONENT TESTERS** Build your own test equipment and save a bundle (and have a lot of fun). Volume I of the 73 Test Equipment Library shows you how to build and use transistor testers (8 of 'em), three diodes testers, 3 IC testers, 9 voltmeters and VTVMs, 8 ohmmeter, 3 inductance meters, and a raft of other gadgets for checking temperature, crystals, Q, etc. $4.95.*

● **VOL II AUDIO FREQUENCY TESTERS** If you're into audio such as digital cassette recording, RTTY, Baudot vs ASCII, SSTV, SSB, Touchtone or even hi-fi you'll want to have this book full of home built test equipment projects. Volume II $4.95.*

● **VOL III RADIO FREQUENCY TESTERS** This is of more interest to hams and CBers: test equipment you can build for checking out transmitters and receivers, signal generators, noise generators, crystal calibrators, GDOs, dummy loads. $4.95.*

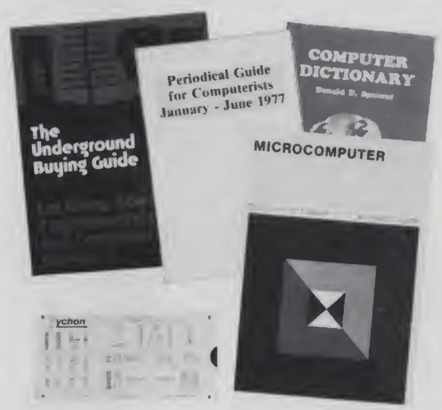
● **VOL. IV IC TEST EQUIPMENT** Become a trouble-shooting wizard. All you need to know about pulse, audio and sync generators, frequency counters, digital component testers, logic probes and more! Plus a cumulative index for all four volumes of the 73 Test Equipment Library. $4.95.*

general

● **COMPUTER DICTIONARY** by Donald D. Spencer. Defines words and acronyms used by computerists in a clear, easy to understand style. This reference is a must for the individual getting started in the world of microcomputers. $5.95.*

● **BRAND NEW DICTIONARY** This microcomputer dictionary fills the need to become quickly acquainted with the terminology and nomenclature of the revolution in computers. There is also a comprehensive electronics/computer abbreviations and acronyms section. $15.95.*

● **THE UNDERGROUND BUYING GUIDE** Here is a handy guide for the electronics enthusiast. Over 600 sources of equipment and literature are provided. Cross-referenced for ease of use. Electronic publishing houses are also listed. $5.95 each.*



● **1976 PERIODICAL GUIDE FOR COMPUTERISTS** is a 20-page book which indexes over 1,000 personal computing articles for the entire year of 1976 from Byte, Creative Computing, Digital Design, Dr. Dobbs Journal, EDN, Electronic Design, Electronics, Interface Age, Microtrek, Peoples Computer Company, Popular Electronics, QST, Radio Electronics, SCCS Interface and 73 Amateur Radio. Price $3.00.* New January – June 1977 Edition (includes Kilobaud) – $3.00.*

● **TYCHON'S 8080 OCTAL CODE CARD** Slide rule-like aid for programming and debugging 8080 software contains all the mnemonics and corresponding octal codes. Also available, Tychon's *8080 Hex Code Card*, same as above only has hex codes instead of octal. $3.00 each.*

amateur radio books

● **NOVICE STUDY GUIDE** The most complete Novice study guide available. It is brand new. This is not only invaluable for anyone wanting to get started in amateur radio, but also it is about the only really simple book on the fundamentals of electricity and electronics. $4.95.*

● **GENERAL CLASS STUDY GUIDE** Takes over on theory where the Novice book leaves off. You'll need to know the electronic theory in this to work with computers and you'll not find an easier place to get the information. $5.95.*

● **SSTV HANDBOOK** This excellent book tells all about it, from its history and basics to the present state-of-the-art techniques. Hardbound $7,* Softbound $5.*

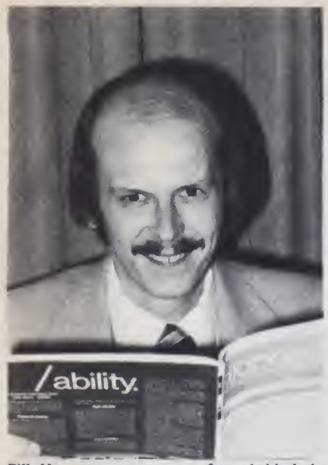
● **VHF ANTENNA HANDBOOK** This new handbook details the theory, design and construction of hundreds of different VHF and UHF antennas. Packed with fabulous antenna projects you can build. $4.95.*

● **WEATHER SATELLITE HANDBOOK** Simple equipment and methods for getting good pictures from the weather satellite. Dr. Taggart WB8DQT $4.95.*

● **THE NEW RTTY HANDBOOK** is a brand new 1977 edition and the only up-to-date RTTY book available. The state-of-the-art has been changing radically and has made all previous RTTY books obsolete. It has the latest circuits, great for the newcomer and the expert alike. $5.95.*

Use the order card in the back of this magazine or itemize your order on a separate piece of paper and mail to:
Kilobaud Book Department Peterborough NH 03458 Be sure to include check or detailed credit card information

***** *Add $1.00 shipping & handling charge for each order.* Note: Prices subject to change on books not published by 73 Magazine.



Bill Honeyman comes out from behind the Kilobaud.

WHO'S BEHIND THE KILOBAUD?

In between selling subscriptions and seeing the other exhibits at computer shows, Wayne has been snapping some pictures of people you have been seeing and will be seeing at shows. How many of them can you recognize behind the Kilobauds?

Kilobaud is the most read magazine in the microcomputer field, so it hasn't been difficult to find key industry people reading the magazine.

On page 139 of the January issue there was a photo of Bill Honeyman of The Digital Group reading Kilobaud. If you get to many of the computerfests or personal computing shows you've talked at length with Bill. He's enthusiastic and fun to talk to. Say hello to Bill at your next show.

kilobaud

READ IT YOURSELF!

If you are not yet a subscriber to Kilobaud, we need you and you need us. There is a lot of data in Kilobaud that you will want to have on hand . . . the magazine is like a continuing encyclopedia of microcomputing and programs. You never know when you are desperately going to need something from a past issue . . . and you'll want it immediately. The cost per year is not significant . . . $15 at present for $24 worth of magazines.

You may have noticed that Kilobaud has more articles than any of the other magazines. If you'll keep track of how long it takes you to read Kilobaud as compared to the other computer magazines, you'll see you are getting a great bargain at $15 per year.

WHO'S BEHIND THE KILOBAUD?

WHO'S THIS?

You've seen this chap a lot too. Just for a hint, his firm makes S-100 bus stuff and one of the most remarkable design engineers in the entire field is deeply involved with the products. They started out with a clock fix circuit . . . got it? Watch next month for the identity of this Kilobaud reader.

SUBSCRIBE!
For Instant Subscriptions Call Our Toll Free Number
1-(800) 358-5473
Have your credit card handy!
Kilobaud • Peterborough NH 03458

why the last bus is the best bus

If you want to be better, you have to be willing to make changes. So, when we started designing our H8 Computer back in 1975, we put aside a lot of pre-conceptions to design a *better* computer. For instance, instead of slavishly adopting an inferior bus as a "standard," we designed our own. We used fifty fully-buffered lines because that's all we really needed with the H8's built-in systems controller. The result? Clean signals, minimum noise and great expansion capabilities.

Next, we picked a male/female connector. Not only is it less expensive than edge connectors, our tests show it's also more reliable! And because it costs less, every H8 motherboard includes all the connectors you need for expansion. There's nothing extra to buy.

The connectors are mounted on the right side of the cabinet at an angle so the boards tilt back. That gives us a couple of significant advantages. First, it allows the boards to be larger without requiring a taller cabinet. So our circuit board layout is more open, which makes the boards easier to build – even if you've never built a kit before. Second, the angled mounting provides easy accessibility to all the components on the face of the boards without an extender. (Many times an extender alters the transmission characteristics of the bus, which masks the problem you're trying to solve or even causes new ones.)

And we plan to support the H8 bus. In the future you can expect to see many new and exciting accessories for the H8 bus like higher density, lower power memories and a variety of unique I/O interfaces.

We're very happy with our bus, and we think you will be, too.

Who minds being different... as long as you're better.

Heathkit Computers

FREE
Heathkit Catalog

Read about nearly 400 money-saving, fun-to-build electronic kits.

Use coupon to send for your mail order catalog or bring coupon to a Heathkit Electronic Center for your catalog.

HEATH
Schlumberger

Heath Company, Dept. 351-380
Benton Harbor, MI 49022

H5

Please send me my FREE Catalog. I am not on your mailing list.

Name__________

Address__________

City__________ State__________

CP-138 Zip__________

AVAILABLE LOCALLY AT HEATHKIT ELECTRONIC CENTERS
(Units of Schlumberger Products Corporation) Retail prices on some products may be slightly higher.

ARIZONA: Phoenix, 85017, 2727 W. Indian School Rd., Phone: 602-279-6247; **CALIFORNIA: Anaheim, 92805,** 330 E. Ball Rd., Phone: 714-776-9420; **El Cerrito, 94530,** 6000 Potrero Ave., Phone: 415-236-8870; **Los Angeles, 90007,** 2309 S. Flower St., Phone: 213-749-0261; **Pomona, 91767,** 1555 Orange Grove Ave. N., Phone: 714-623-3543; **Redwood City, 94063,** 2001 Middlefield Rd., Phone: 415-365-8155; **Sacramento, 95825,** 1860 Fulton Ave., Phone: 916-486-1575; **San Diego (La Mesa, 92041),** 8363 Center Dr., Phone: 714-461-0110; **San Jose (Campbell, 95008),** 2350 S. Bascom Ave., Phone: 408-377-8920; **Woodland Hills, 91364,** 22504 Ventura Blvd., Phone: 213-883-0531; **COLORADO: Denver, 80212,** 5940 W. 38th Ave., Phone: 303-422-3408; **CONNECTICUT: Hartford (Avon, 06001),** 395 W. Main St. (Rte. 44), Phone: 203-678-0323; **FLORIDA: Miami (Hialeah, 33012),** 4705 W. 16th Ave., Phone: 305-823-2280; **Tampa, 33614,** 4019 West Hillsborough Ave., Phone: 813-886-2541; **GEORGIA: Atlanta, 30342,** 5285 Roswell Rd., Phone: 404-252-4341; **ILLINOIS: Chicago, 60645,** 3462-66 W. Devon Ave., Phone: 312-583-3920; **Chicago (Downers Grove, 60515),** 224 Ogden Ave., Phone: 312-852-1304; **INDIANA: Indianapolis, 46220,** 2112 E. 62nd St., Phone: 317-257-4321; **KANSAS: Kansas City (Mission, 66202),** 5960 Lamar Ave., Phone: 913-362-4486; **KENTUCKY: Louisville, 40243,** 12401 Shelbyville Rd., Phone: 502-245-7811; **LOUISIANA: New Orleans (Kenner, 70062),** 1900 Veterans Memorial Hwy., Phone: 504-722-6321; **MARYLAND: Baltimore, 21234,** 1713 E. Joppa Rd., Phone: 301-661-4446; **Rockville, 20852,** 5542 Nicholson Lane, Phone: 301-881-5420; **MASSACHUSETTS: Boston (Peabody, 01960),** 242 Andover St., Phone: 617-531-9330; **Boston (Wellesley, 02181),** 165 Worcester Ave. (Rt. 9 just west of Rt. 128), Phone: 617-237-1510; **MICHIGAN: Detroit, 48219,** 18645 W. Eight Mile Rd., Phone: 313-535-6480; **E. Detroit, 48021,** 18149 E. Eight Mile Rd., Phone: 313-772-0416; **MINNESOTA: Minneapolis (Hopkins, 55343),** 101 Shady Oak Rd., Phone: 612-938-6371; **MISSOURI: St. Louis (Bridgeton), 63044,** 3794 McKelvey Rd., Phone: 314-291-1850; **NEBRASKA: Omaha, 68134,** 9207 Maple St., Phone: 402-391-2071; **NEW JERSEY: Fair Lawn, 07410,** 35-07 Broadway (Rte. 4), Phone: 201-791-6935; **Ocean, 07712,** 1013 State Hwy. 35, Phone: 201-775-1231; **NEW YORK: Buffalo (Amherst, 14226),** 3476 Sheridan Dr., Phone: 716-835-3090; **Jericho, Long Island, 11753,** 15 Jericho Turnpike, Phone: 516-334-8181; **Rochester, 14623,** 937 Jefferson Rd., Phone: 716-244-5470; **White Plains (North White Plains, 10603),** 7 Reservoir Rd., Phone: 914-761-7690; **OHIO: Cincinnati (Woodlawn, 45215),** 10133 Springfield Pike, Phone: 513-771-8850; **Cleveland, 44129,** 5444 Pearl Rd., Phone: 216-886-2590; **Columbus, 43229,** 2500 Morse Rd., Phone: 614-475-7200; **Toledo, 43615,** 48 S. Byrne Rd., Phone: 419-537-1887; **PENNSYLVANIA: Philadelphia, 19149,** 6318 Roosevelt Blvd., Phone: 215-288-0180; **Frazer (Chester Co.), 19355,** 630 Lancaster Pike (Rt. 30), Phone: 215-647-5555; **Pittsburgh, 15235,** 3482 Wm. Penn Hwy., Phone: 412-824-3564; **RHODE ISLAND: Providence (Warwick, 02886),** 558 Greenwich Ave., Phone: 401-738-5150; **TEXAS: Dallas, 75201,** 2715 Ross Ave., Phone: 214-826-4053; **Houston, 77027,** 3705 Westheimer, Phone: 713-623-2090; **VIRGINIA: Alexandria, 22303,** 6201 Richmond Hwy., Phone: 703-765-5515; **Norfolk (Virginia Beach, 23455),** 1055 Independence Blvd., Phone: 804-460-0997; **WASHINGTON: Seattle, 98121,** 2221 Third Ave., Phone: 206-682-2172; **WISCONSIN: Milwaukee, 53216,** 5215 W. Fond du Lac, Phone: 414-873-8250.

kilobaud

index of advertisers

peterborough nh 03458

kilobaud reader service

Circle appropriate Reader Service # for desired company brochures, data sheets or catalogs and mail to Kilobaud Magazine, Peterborough NH 03458. Include your zip code, please. Send money directly to advertisers. LIMIT: 25 requests.

	C53	C59	E21	J6	M33	O7	P21	S40	T27
A37	C41	C34	F4	J1	M28	O2	P27	S5	V14
A49	C36	C31	G4	L3	M31	O3	Q3	S6	W19
A38	C28	C60	H20	M21	P11	O8	R12	T11	W13
A48	C33	C57	H5	M17	M7	O6	R14	T8	W16
A43	C30	D13	I13	M27	M32	P9	R16	T26	KILOBAUD
B21	C48	D12	I9	M16	N8	P7	R7	T13	
B18	C44	D20	I10	M30	O5	P24	S2	T1	

PLEASE PRINT OR TYPE ☐ Newsstand buyer ☐ Subscriber

This Month's ARTICLE WINNER: (Title page number) ____________

Name ____________

Address ____________

City ____________ State ________ Zip ________

Coupon Expires in 60 Days

2/78

PLACE STAMP HERE

kilobaud

peterborough nh 03458

Att: Mail Order

I want to subscribe!

Send me Kilobaud for:

☐ 1 year — $15.00*

☐ 3 years — $36.00**

☐ Life Subscription — $155.00**

☐ Renewal

☐ New sub.

Enclosed $________ ☐ Cash ☐ Check ☐ Money Order

☐ American Express ☐ BankAmericard ☐ Master Charge

Credit card # ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐

Expiration date ________ Signature ____________

☐ Bill me directly Signature ____________

Name ____________

Address ____________

City ____________ State ________ Zip ________

* U.S. & Canada only. Others write for foreign rates.

**U.S. & Canada only.

Subscription prices expire in 60 days.

2/78

FIRST CLASS
Permit No. 1024
Peterborough NH 03458

BUSINESS REPLY MAIL

NO POSTAGE NECESSARY IF MAILED IN UNITED STATES

POSTAGE WILL BE PAID BY

kilobaud

PETERBOROUGH NH 03458

Att: Reader Service

PLEASE PRINT OR TYPE   Please send me the following Kilobaud products:

Quantity	Description	Unit Price	Total

Total ______

Shipping & Handling Charge $1.00

Order Total ______

Enclosed $_____ ☐ Cash ☐ Check ☐ Money Order ☐ C.O.D.

Bill: ☐ American Express ☐ BankAmericard ☐ Master Charge

Credit Card # ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐

Expiration date ________ Signature ____________________

Name ____________________

Address ____________________

City ____________________ State ______ Zip ________ 2/78

PLACE STAMP HERE

kilobaud

PETERBOROUGH NH 03458

Att: Subscriptions

books,etc.

- BASIC $4.95
- Basic Software Library, Vol I & II $24.95 ea., Vol III $39.95, Vol IV & V $9.95 ea.
- Brand New Dictionary $15.95
- Build-It Book of Digital Electronic Timepieces $6.95
- CMOS Cookbook $9.95
- Chemistry With a Computer $9.95
- The Compulator Book $7.95
- Computer Dictionary $5.95
- Computer Programming Handbook $8.95
- Discovering BASIC $6.85
- Fortran Programming $7.95
- Fortran Workbook $3.95
- Fun With Computers and BASIC $6.95
- General Class Study Guide $5.95
- Hobby Computers Are Here! $4.95
- Home Computers, 2^{10} Q&A, Vol I $7.95, Vol II $6.95
- An Introduction to Microcomputers, Vol I $7.50, Vol II $12.50
- Introduction to Microprocessors $17.50 USA, $20.00 elsewhere
- Microcomputer Dictionary $15.95
- Microcomputer Primer $7.95
- Microprocessor Interfacing Techniques $9.95
- Microprocessor Programming $8.95
- Microprocessors $9.95
- My Computer Likes Me . . . When I speak BASIC $2.00
- The New Hobby Computers $4.95
- Novice Study Guide $4.95
- 1976 Periodical Guide for Computerists $3.00 January-June 1977 Edition $3.00

kilobaud

- A Quick Look at BASIC $4.95
- The New RTTY Handbook $5.95
- Scelbi's First Book of Computer Games $14.95
- Scelbi's Galaxy Game for the "6800" $14.95
- Secret Guide to Computers, Part I $2.75, Part II $2.50, Part III $3.50
- Some Common BASIC Programs $7.50
- Spencer's Computer Dictionary $5.95
- SSTV Handbook $5.00 (Soft), $7.00 (Hard)
- The Story of Computers $4.95
- TTL Cookbook $8.95
- TVT Cookbook $9.95
- Test Equipment Library $4.95 ea.
  - Vol I Component Testers
  - Vol II Audio Frequency Testers
  - Vol III Radio Frequency Testers
  - Vol IV IC Test Equipment
- Tychon's 8080 Hex Code Card $3.00
- Tychon's 8080 Octal Code Card $3.00
- The Underground Buying Guide $5.95
- VHF Antenna Handbook $4.95
- Weather Satellite Handbook $4.95
- What to Do After You Hit Return $8.00
- 60 Challenging Problems With BASIC Solutions $6.95
- 101 BASIC Computer Games $7.50
- 6800 Programming for Logic Design $7.50
- 6800 Software Gourmet Guide and Cookbook $9.95
- 8080 Programming for Logic Design $7.50
- 8080 Software Gourmet Guide and Cookbook $9.95

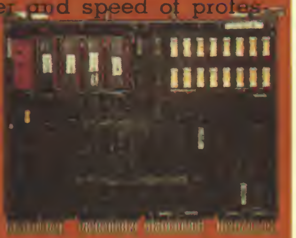
KB Back Issues – Issue #1 – $5.00 ea.; Others – $3.00 ea.

Binders for Kilobaud 1977 – $6.00 ea.; 2 for $11.00, $5.00 each additional binder.

CM T-shirts $5.50 ea.

the world's most powerful microcomputer comes home

Computer hobbyists have always wanted the power and speed of professional machines. But they've had to settle for less. Professional machines were too expensive. Not anymore. Now there's the Heathkit H11.



Professional performance, kit price. The H11 uses Digital Equipment Corporation's 16-bit LSI-11 CPU. Exactly the same CPU found in the famous DEC PDP-11 – probably the most popular commercial and industrial computer in the world. So now a low-cost kit gives you the speed, power and throughput of the world's most popular professional computer.

Software and support. As an H11 owner, you have access to an incredible range of software – editor, relocatable assembler, linker, absolute loader, debug program, I/O executive program, dump routines, BASIC and FOCAL. And, by joining the DEC user's group (DECUS), you can have access to one of the largest software libraries in the world. And there's more to come. Much more.

Of course, the H11 executes the powerful PDP 11/40 instruction set which includes over 400 commands. And the fully assembled KD-11F CPU board includes 4K x 16 bits of memory. The backplane accepts up to six additional modules (memory, serial and parallel I/O, etc.)

Heath will soon introduce a dual floppy with a disk operating system to take full advantage of the H11's power. And the H11 is fully compatible with all DEC LSI-11 accessories.

Heath documentation is second to none. You get illustrated step-by-step instructions on how to build the kit, thorough explanations of the software and comprehensive operating instructions. If you want to see for yourself, we'll send you the entire manual set (#HM-1100) for just $25. And you can apply the cost to the price of the computer.

With the Heathkit H11, you can enjoy power, speed, versatility and potential no 8-bit machine can match. And the H11 costs little more than lesser computers – just $1295.*

Don't settle for less than the best. A close look in the latest Heathkit catalog will convince you that the H11 is the best home computer you can buy.

DEC, DECUS, PDP and FOCAL are registered trademarks of Digital Equipment Corp.

*Mail order, FOB Benton Harbor Michigan. Retail price slightly higher.

Prices and specifications subject to change without notice.

HEATHKIT COMPUTERS

System Engineered for Personal Computing

H11

Be sure to use coupon on facing page of this magazine to order your FREE Heathkit Catalog!

Jim-pak electronic components

One-Stop Component Center

QUALITY PRODUCTS

TTL

7400	7490
7402	7492
7404	7493
7408	74107
7410	74121
7420	74122
7430	74123
7432	74125 (DM8093)
7442	74154
7447	74161 (DM9316)
7473	74176 (DM8280)
7474	74177 (DM8281)
7475	74192
7476	74193
7485	74367 (DM8097)

Low Power TTL Schottky

74LS00	74LS83
74LS02	74LS85
74LS04	74LS86
74LS08	74LS90
74LS10	74LS109
74LS20	74LS123
74LS30	74LS136
74LS32	74LS138
74LS73	74LS151
74LS74	74LS175
74LS75	74LS367

C/MOS

4000	4029
4001	4044
4010	4046
4011	4049
4013	4050
4016	4051
4017	4069
4020	4071
4023	4081
4024	4511

LINEAR

LM301V	LM340T 15(7815)
LM307V	LM555V
LM309K	LM556N
LM311V	LM567V
LM320K 5(7905)	LM723N
LM320T-5(7905)	LM739N
LM320T 12(7912)	LM741V
LM320T-15(7915)	LM747N
LM339N	LM1458V(5558V)
LM340T 5(7805)	LM1556V
LM340T 12(7812)	LM3900N(CA3401)

MICROPROCESSOR

Z 80	2101
8080A	2102
8212	21L02
8224	7489
8228	MM5262
6800	1702A
6810	82S23
6830	2708
AY 5-1013	DM8835N
2513/2140	N8T97

DIODES

IN751	IN4148(IN914)
IN4733	IN4001
IN4734	IN4004
IN4742	IN4007
IN4744	MDA 980 3

TRANSISTORS

C10681	2N3055
2N2222A	2N3904
2N2907A	2N3906
MJE2955	2N5129
MJE3055	2N5139

RESISTORS

ASST 1	ASST 5
ASST 2	ASST 6
ASST 3	ASST 7
ASST 4	

CAPACITORS

Aluminum Electrolytic

1mfd 50V	100mfd 50V
4 7mfd 50V	220mfd 50V
10mfd 50V	470mfd 50V
22mfd 50V	1000mfd 25V
47mfd 50V	2200mfd 16V

Ceramic Disc

10pf 50V	001mfd 50V
47pf 50V	0047mfd 50V
100pf 50V	01mfd 50V
220pf 50V	022mfd 50V
330pf 50V	047mfd 50V
470pf 50V	1mfd 50V

Dipped Tantalum

1mfd 35V	2 2mfd 25V
22mfd 35V	3 3mfd 25V
33mfd 35V	4 7mfd 25V
47mfd 35V	6 8mfd 25V
68mfd 35V	10mfd 25V
1mfd 35V	15mfd 25V
1 5mfd 35V	33mfd 25V

SWITCHES

SPST 4 sta dip	SPDT push button
SPST 8 sta dip	SPDT push button
SPDT on-off on toggle	SPST push button
	SPST push button
SPDT on-none on toggle	SPST push button
	DPDT slide
DPDT on-off-on toggle	DPDT on-none-on toggle

SOCKETS

8 pin low profile	14 pin wire wrap
14 pin low profile	16 pin wire wrap
16 pin low profile	24 pin wire wrap
24 pin low profile	40 pin wire wrap
40 pin low profile	TO-3 Socket
Molex Pins	TO 5 Socket

Display LEDS Discrete

5×7 dot matrix	600 com anode
300 com anode	600 com cathode
300 com cathode	200LED Lamp-red

CONNECTORS

D8 25P Plug	D825S Socket

DATA BOOKS*

7400/74LS Data Book
CMOS/Linear Data Book
Microprocessor/LED Data Book
*JIM PAK Products only

A component product line developed for the independent dealer. Guaranteed, nat lly advertised products. Complete **JIM-PAK** program includes national advertising, direct mail programs, store display cks, stock rotation plan and return policy. For dealer information, write or call **JIM-PAK**, 1021 Howard Avenue, Sa arlos, CA 94070 (415) 592-8097.

★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★

SEE YOUR LOCAL Jim-pak DEAL R TODAY...

ALABAMA
Mobile — *Lafayette Radio Electronics*

ARIZONA
Fountain Hills — *P & C Communications*

CALIFORNIA
Bellflower — *Earl's Hobby Shop*
Berkeley — *Al Lasher Electronics*
Fontana — *Fontana Electronics*
Long Beach — *Scott Radio Supply*
Monterey — *Zackit*
Palo Alto — *Zack Electronics*
Pasadena — *Dow Radio Inc.*
Sacramento — *The Radio Place*
Sacramento — *Zackit*
San Carlos — *J&H Outlet Store*
San Francisco — *Zack Electronics*
San Jose — *Quement Electronics*
Sunnyvale — *Sunnyvale Electronics*
Vallejo — *Zackit*
Walnut Creek — *Byte Shop Computer Store*

CANADA
Alberta (Calgary) — *The Computer Shop*

COLORADO
Aurora — *Com Co Electronics*
Steamboat Springs — *Norm's TV & Electronics*

CONNECTICUT
Bridgeport — *Computer World*

FLORIDA
Lakeland — *Lakeland Specialty Electronics*
Orlando — *Altair Computer Center of Orlando*
Tampa — *AMF Electronics*
Tampa — *Microcomputer Systems*

FRANCE
Paris — *Computer Boutique*

GEORGIA
Atlanta — *Atlanta Computer Mart*

HAWAII
Aiea — *Delcoms Hawaii*
Honolulu — *Integrated Circuit Supply*

IDAHO
Idaho Falls — *Audiotronics*

ILLINOIS
Evanston — *Itty Bitty Machine Co.*
Groveland — *Moyer Electronics*

ILLINOIS (Cont'd)
Mount Prospect — *Tri-State Electronics*
Oak Park — *Spectronics*

INDIANA
East Chicago — *Acro Electronics*
Hammond — *Quantum Computer Works*

KENTUCKY
Lexington — *Radio-Electronic Equipment Co.*

LOUISIANA
Baton Rouge — *Davis Electronics Supply*

MARYLAND
Baltimore — *Computer Workshop of Baltimore*
Baltimore — *Everything Electronic*
Rockville — *Computer Workshop*
Towson — *Computers, Etc.*

MASSACHUSETTS
Medford — *Tufts Electronics*
North Adams — *Electronics Supply Center*
Waltham — *Computer Mart Inc.*

MICHIGAN
Grand Rapids — *Micro Computer World*
Lansing — *Fulton Radio Supply*

MINNESOTA
Duluth — *Northwest Radio of Duluth*
Eagan — *Dacom Amateur Radio Center*

MISSOURI
Parkville — *Computer Workshop of Kansas City*

MONTANA
Billings — *Conley Radio Supply*

NEBRASKA
Lincoln — *Altair Computer Center*
Omaha — *Omaha Computer Store*

NEVADA
Las Vegas — *Century 23*

NEW JERSEY
Cherry Hill — *Computer Systems Unlimited*
Hoboken — *Hoboken Computer Works*
Pompton Lakes — *Computer Corner of New Jersey*
Ramsey — *Typetronic Computer Store*

NEW YORK
Albany — *Fort Orange Electronics*
New York — *The Computer Stores Inc.*
New York — *Computer Mart of New York*
Troy — *Trojan Electronics*
White Plains — *The Computer Corner*

NORTH CAROLINA
Raleigh — *Byte Shop of Raleigh*

IIO
Bucyrus — *Mead Electronics*
Cincinnati — *Digital Design*
Dayton — *Altair Computer Center*
Reynoldsburg — *Universal Amateur Radio*
Steubenville — *Hosfelt Electronics*

LAHOMA
Guymon — *Sound Service*
Oklahoma City — *Bits, Bytes & Micros*

EGON
Beaverton — *Altair Computer Center*
Coos Bay — *Herrick Electronix*
Ontario — *Miller Electronics*

NAMA
anama City — *Sonitel, S.A.*

NNSYLVANIA
Hershey — *Microcomputer Systems Inc.*
Murraysville — *Computer Workshop of Pittsburgh*

RHODE ISLAND
Cranston — *Jabbour Electronics City*
Pawtucket — *Jabbour Electronics City*

SOUTH CAROLINA
No. Charleston — *Technical Services Inc.*

TENNESSEE
Knoxville — *Byte Shop*
Memphis — *Sere-Rose & Spencer Electronics*
Oak Ridge — *Computer Denn*

TEXAS
Dallas — *Computer Shops Inc.*
Houston — *Altair Computer Center*
Houston — *Interactive Computers*
San Antonio — *Sherman Electronics Supply*

IRGINIA
Alexandria — *Computer Hardware Store*
Alexandria — *Computers Plus*
Richmond — *Computers-To-Go*
Springfield — *Computer Workshop of North Virginia*
Virginia Beach — *Heathkit Electronics Center*

WASHINGTON
Bellevue — *Altair Computer Center*
Longview — *Progress Electronics*
Pasco — *Riverview Electronics*
Spokane — *Personal Computers*

WEST VIRGINIA
Morgantown — *The Computer Corner*
Morgantown — *Electro Distributing Co.*